UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

Geração Automática de Conhecimento para SDI extraído de OSINTs

Ivo Ricardo Guerreiro Vacas

Mestrado em Segurança Informática

Dissertação orientada por:
Prof. Doutora Ibéria Vitória de Sousa Medeiros e Prof. Doutor Carlos Nuno da Cruz Ribeiro

2017

Agradecimentos

Este espaço é dedicado a quem, para além de mim, tornou esta dissertação possível:

À minha família que, apesar de não perceber de grande coisa do que faço, teve a confiança (ou desleixo) de me facultar tudo o que foi necessário para terminar a coisa.

À minha professora e orientadora Ibéria Medeiros por me ter lembrado a um mês da entrega do relatório preliminar, que ainda só tinha feito a inscrição e que isso não era suficiente, por ter sido presente, construtiva, assertiva e melhor que isso, amiga.

Ao Miguel Cabral por ter transposto para a minha pessoa a experiência que só se consegue ao ter uma tese que dura há dois anos, à sua magia negra nos scripts, pela sua especialização em data analitics no notepad++, MS Word e Excel, pela companhia ao longo de todo o percurso e por não me ter deixado engordar sozinho.

Aos meus amigos gabo, para além da paciência e (pouca) compreensão da minha constante ausência, a criatividade para me adjetivarem com nomes de animais, como por exemplo, toupeira.

Aos meus colegas e amigos do Departamento de Informática da Reitoria da Universidade de Lisboa que sempre me apoiaram e que de tudo fizeram para que não me faltassem os recursos necessários para que o meu percurso como estudante e profissional se realizasse com sucesso. Entre todo o espetacular Departamento de Informática, as pessoas que mais diretamente contribuíram dão-se pelo nome de André Brioso, Luís Caldeira, Nuno Abrantes e Pedro Pereira.

Um obrigado à Universidade de Lisboa por ter proporcionado toda a minha aprendizagem e por ter sido o palco da minha primeira experiência profissional.

Por fim, este trabalho foi parcialmente suportado pela EC através do projeto H2020-700692 (DiSIEM) e pelos fundos nacionais através da Fundação para a Ciência e a Tecnologia (FCT) com referência para UID/CEC/00408/2013 (LaSIGE).

Resumo

A crescente evolução das tecnologias de informação e comunicação, aplicações e serviços fez com que, nos dias de hoje, o recurso a computadores e à comunicação entre eles se tornasse imprescindível para a realização de enumeras ações e de diversos fins, desde o acesso a dados até a transações financeiras. A par desta evolução, o cibercrime tem vindo a intensificar-se, tendo como intuito a apropriação ilícita de dados privados e de monetização. Os ataques informáticos atuais praticados no cibercrime são mais sofisticados, de tal forma que a sua deteção é cada vez mais difícil e os seus danos são cada vez mais devastadores. Este facto leva a que o cibercrime seja uma preocupação tanto para as empresas como para os profissionais de segurança informática.

Um dos recursos utilizados para a concretização destes ataques são as *botnets*. As botnets são redes compostas por dispositivos vulneráveis (*bots*), tais como computadores, e controladas por entidades criminosas. A sua utilização permite o anonimato destas entidades durante os ataques. Para se protegerem destes ataques, as organizações utilizam mecanismos de defesa, tais como *sistemas detetores de intrusões* (SDI), no entanto, a sua eficácia na deteção destes ataques depende do conhecimento que estes contêm sobre as ameaças e da forma como as deteta, tornando obrigatório que os SDI sejam atualizados constantemente com conhecimento sobre novas ameaças. Este conhecimento pode ser obtido de diversas fontes de inteligência - *Open Source Intelligence* (OSINT) - públicas, as quais se encontram acessíveis em diversos locais na Internet.

Esta dissertação apresenta uma solução para melhorar uma arquitetura de deteção de intrusões em SDI para detetar bots dissimulados na rede de computadores da Universidade de Lisboa. A solução propõe um *gerador de regras e blacklists* para SDI, com base em informação OSINT recolhida por uma plataforma de threat intelligence, bem como a integração automática das regras e blacklists no SDI.

Foi realizada uma avaliação experimental da solução em ambiente real, usando 44 fontes de OSINT coletadas pela plataforma threat intelligence IntelMQ e o SDI Snort. A arquitetura proposta permitiu detetar ameaças de diversas categorias.

Palavras-chave: deteção de ameaças, SDI, OSINT, threat intelligence, segurança

Abstract

The continuous evolution of information and communication technologies, applications and services, made the computer an essential resource for performing several and different tasks, such as access to data and financial transactions. Alongside this evolution, the cybercrime has been increasing and unfortunately a common action. Its purpose is the illegal appropriation of private data, and monetization. The cyber-attacks became more sophisticated, in such a way that their detection is getting more difficult and their damage increasingly devastating. This makes the cybercrime a concern for both enterprises and security professionals.

One of the most well-known techniques used to practice these crimes, is the creation and management of *botnets*. Botnets are networks composed by vulnerable devices (*bots*) such as computers, and controlled by criminal entities. Its use allows the anonymity of these entities when the attacks are performed.

Enterprises, for protecting themselves against these attacks, they use defence mechanisms, such as *intrusion detection systems* (IDS), however, their effectiveness in detecting these attacks depends on their knowledge about threats and how they detect them, turning mandatory that IDSs must be constantly updated with knowledge of new threats. This knowledge can be obtained from many public intelligence sources - *Open Source Intelligence* (OSINT), which are accessible at several locations on the Internet.

The main objective of this dissertation is the improvement of an intrusion detection architecture by using a IDS to detect hidden and dissimulated bots in the network infrastructure of the University of Lisbon. The presented solution proposes a *rule and blacklist generator* for IDS, using information from OSINT feeds collected and processed by a threat intelligence platform, and the automatic integration of the rules and blacklists in the IDS.

An experimental evaluation of the solution in a real environment was performed using 44 OSINT sources collected by the IntelMQ threat intelligence platform, and the Snort IDS. The proposed architecture detected threats of several categories.

Keywords: threat detection, IDS, OSINT, threat intelligence, security

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

C&C	Command & Control Center
CPU	Central Processing Unit
CERT	Computer Emergency Response Team
DB	Database
DDoS	Distributed Denial of Service
DNS	Domain Name System
DST	Destionation
EUA	Estados Unidos da América
FCCN	Fundação para a Computação Científica Nacional
FTP	File Transfer Protocol
GB	Gigabytes
Gbps	Gigabit per second
GHz	Giga Hertz
HP	Hewlet Packard
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IDES	Intrusion Detection Expert System
IEEE	Instituto de Engenheiros Eletricistas e Eletrónicos
IMAP	Internet Message Access Protocol
IES	Integrated Engineering Software
IOA	Indicator of Attack
IoE	Internet of Everything
IOC	Indicator of Compromise
IoT	Internet of Things
IP	Internet Protocol
IRC	Internet Relay Chat
JSON	JavaScript Object Notation
KISS	Keep It Simple, Stupid
MISP	Malware Information Sharing Platform
MQ	Messaging Queue

MS	Microsoft
NSF	National Science Foundation
OISF	Open Information Security Foundation
OSINF	Open Source Information
OSINT	Open Source Intelligence
P2P	Peer to Peer
POSIX	Portable Operating System Interface
RAM	Random Access Memory
RPS	Requests Per Second
SDI	Sistema Detetor de Intrusões
SGBD	Sistema de Gestão de Base de Dados
SIP	Session Initiation Protocol
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Manager Protocol
SRC	Source
SSDP	Simple Service Discovery Protocol
SSH	Secure Shell
SSL	Secure Socket Layres
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UL	Universidade de Lisboa
URL	Uniform Resource Locator
VoIP	Voice over Internet Protocol
WiSM	Wireless Service Module

Capítulo 1

Introdução

A crescente evolução das tecnologias de informação e comunicação, aplicações e serviços fez com que, nos dias de hoje, o recurso a computadores e à comunicação entre eles se tornasse imprescindível para a realização de enumeras ações e de diversos fins, desde o acesso a dados até a transações financeiras. Esta evolução permitiu também a comunicação entre dispositivos móveis, originando a IoT (*Internet of Things*) e a presentemente IoE (*Internet of Everything*), que permitem interligar milhões desses dispositivos.

Os objetivos atuais do cibercrime não são muito diferentes daqueles que eram uma década atrás. Agora, tal como nessa altura, o objetivo principal do cibercrime é a obtenção de vantagens económicas para o atacante, sobre o alvo vitima. O que difere são as técnicas de monetização, em número e sofisticação, para a realização destes crimes. Enquanto anteriormente a monetização de um crime passava usualmente pelo roubo de um cartão de crédito, atualmente existem empresas de serviços (entidades criminosas) cujo o único propósito é monetizar o cibercrime. A par da evolução tecnológica, o cibercrime tem vindo a intensificar-se nos últimos anos, sendo cada vez mais uma preocupação tanto para as empresas como para os profissionais de segurança informática [1]. Os ataques cibernauticos atuais são mais sofisticados, de tal forma que a sua deteção é cada vez mais difícil e os seus danos devastadores.

Um dos recursos utilizados para a realização destes ataques são as *botnets*. As botnets são redes de dispositivos vulneráveis, tais como computadores e dispositivos móveis, apelidados de *bots*, e controladas por entidades criminosas através de um centro de comando e controlo (C&C) [2]. O uso de botnets permite que essas entidades realizem ataques, aproveitando-se de recursos que não lhes pertencem e mantendo-se anónimas [1].

Os ataques a servidores organizacionais com o suporte de botnets, tais como o Distributed Denial of Service (DDoS), spamming e propagação de malware, são cada vez mais frequentes. Segundo o relatório de 2016 da Akamai [3], estes ataques têm aumentado, tendo havido um aumento de 9% do primeiro para o segundo quadrimestre de 2016 e de 129% comparativamente com o ano anterior. O tipo de ataque mais prevalente foi o de DDoS e os setores económicos mais afetados foram os financeiros e retalho [3]. A Nitol e a Mirai botnets [4] [5] foram duas das botnets descobertas em 2016 e que estiveram na base destes números. Por exemplo, o maior ciber-ataque realizado com a Mirai botnet foi o massivo Dyn DNS DDoS contra os servidores da Dyn, a organização que controla muitos dos servidores de DNS da Internet. No ataque participaram aproximadamente 500.000 dispositivos IoT tendo gerado um volume de tráfego na ordem dos 1.2 Tbps, provocando uma interrupção do serviço de Internet em toda a Europa e EUA [5] [6]. A grandeza e a frequência de tais ataques são preocupantes e não contrariam as previsões de ciber segurança que foram realizadas para o ano de 2016 e que se mantêm para 2017 [7] [8]. Prevê-se que até ao ano de 2019 os danos de tais ataques atinjam 2 triliões de dólares [9].

O C&C das botnets é um alvo particularmente apetecível, quer para as autoridades que tentam destruir as atividades por elas realizadas, quer para outros ciber-criminosos que pretendem apropriar-se deles para poderem usa-los em seu proveito. As entidades criminosas utilizam mecanismos criptográficos para protegerem a comunicação nas botnets e mecanismos de dissimulação para impedirem a sua identificação e desmantelamento [10], evitando assim a sua captura por sistemas de monitorização de redes de computadores. Por exemplo, um mecanismo eficaz de dissimulação de bots é o de metamorfose, que permite que estes sejam difíceis de detetar por tais sistemas de deteção devido à sua constante mudança comportamental [11].

Face a este problema, as organizações para se protegerem utilizam *sistemas detetores de intrusões* (SDI) que monitorizam o tráfego da rede, detetando anomalias e gerando alertas destas. Com base nos alertas as equipas de segurança analisam os incidentes e executam medidas de proteção. O modo primordial de funcionamento dos SDI é baseado no conhecimento do comportamento ou de assinaturas, que quando há um desvio de comportamento ou igualdade de assinatura, o SDI gera um alerta [12]. No entanto, estes sistemas são conhecidos por reportarem falsos positivos (alertas de falsas anomalias) devido a ligeiros desvios comportamentais que possam ocorrer, ao contrário

dos falsos negativos (anomalias não detetadas) devido a não possuírem o conhecimento suficiente sobre todas as anomalias e ataques existentes [13].

Apesar da importância dos SDIs na deteção de ataques, nomeadamente naqueles que utilizam botnets, existe a necessidade constante de fornecer conhecimento aos SDIs, pois novos e sofisticados ataques surgem muito rapidamente, deixando os SDIs desatualizados. A obtenção deste conhecimento, por um lado, é considerada privada e segredo de negócio [14], não estando, portanto, acessível a todos. Por seu turno, este conhecimento pode ser obtido de diversas fontes de inteligência – *Open Source Intelligence* (OSINT) – públicas e acessíveis em diversos locais na Internet. Os OSINTs contêm informação partilhada que provém de honeypots e de eventos reais que causaram algum tipo de dano em informação ou serviços, como é o caso de roubo de credenciais através de Phishing ou ataques de DDoS [12] [13].

1.1 Motivação

Os servidores organizacionais têm de possuir um poder computacional grande o suficiente quanto o número de máquinas clientes que têm de servir, por forma a serem capazes de responder a pedidos em grande escala e sem comprometerem a disponibilidade do serviço que oferecem. A título de exemplo, o Twitter em 2009, quando tinha aproximadamente 350.000 utilizadores, respondia a 600 RPS (Requests Per Second), sendo que os seus sistemas de gestão de base de dados (SGBD) respondiam a 2400 RPS [15]. Este tipo de servidores são sistemas altamente apetecíveis para os ciber-criminosos porque, caso os consigam controlar, conseguem praticar ataques perigosos e em grande escala. Ataques contra estes servidores são frequentes, sendo os mais usuais o DDoS, spamming e propagação de malware, que recorrem ao uso de botnets [16].

De forma a manter o anonimato e expandir os recursos computacionais para a prática de cibercrimes, os criminosos apropriam-se de máquinas vítimas. Para tal, os criminosos geralmente instalam nestas máquinas *malware* ou *backdoors* para posteriormente controlá-las e geri-las, remotamente. Estas máquinas comprometidas são denominadas de bots e as redes que elas formam são apelidadas de botnets.

A Universidade de Lisboa (UL) é uma instituição internacionalmente reconhecida e alberga um elevado número de alunos, tanto nacionais, como do programa Erasmus. De forma a poder concretizar os serviços prestados aos seus alunos, os sistemas informáticos da universidade precisam de responder a um número elevado de pedidos em larga escala,

o que tal só é possível com servidores com uma grande disponibilidade e poder computacional. Este facto torna a rede de computadores da instituição num alvo apetecível a ciber-criminosos. Um dos serviços oferecidos pela universidade que torna a sua rede mais aliciante aos ciber-criminosos, é o serviço de rede sem fios, ou seja, o acesso à rede *eduroam* (Education Roaming). Este serviço permite aos utentes da instituição, como os alunos, conectarem-se à rede de computadores da instituição para usufruírem dos serviços que esta oferece. Uma vez que os computadores pessoais dos alunos que se ligam através deste serviço não são monitorizados, a probabilidade de haver máquinas infetadas é muito elevada.

A largura de banda da instituição, aliada ao serviço eduroam e aos computadores não monitorizados dos alunos, coloca a rede da universidade com um risco muito elevado. Tendo isto em conta, a instituição necessita de uma ferramenta de segurança que permita identificar e informar, em tempo real, ameaças existentes nos servidores e nos computadores laborais de toda a equipa por detrás da concretização desses serviços, bem como na rede eduroam. Nesta última é de onde provavelmente se encontra a maior quantidade de ameaças.

1.2 Objetivos

Este trabalho tem como intenção o estudo e implementação de mecanismos de deteção de ameaças no tráfego da rede, tais como botnets. Através de um levantamento das atuais soluções de SDI, conjuntamente com as tendências criminais deste tipo de ameaças e com a informação provida de OSINT, será criada uma solução que permita combater este problema na UL. É necessário que este mecanismo de deteção funcione de forma automática, por forma a resolver os possíveis problemas de ciber-segurança existentes nos serviços centrais e na rede eduroam da UL. Assim sendo, esta dissertação tem os seguintes objetivos:

- Compreender como são caracterizadas as botnets, os mecanismos de dissimulação destas, e como identificar tráfego produzido por elas. Estudar outras ameaças/ataques que podem ser realizados recorrendo a botnets. Simultaneamente, são estudados SDI's por forma a compreender o seu funcionamento e processamento de regras, visando assim, a geração automática destas e a sua integração no SDI.

- Estudar fontes OSINT que contenham informação suficientes e necessárias para caracterizar ameaças e convertê-las em regras de SDI.
- Apresentar uma solução para melhorar uma arquitetura de deteção de intrusões em SDI. A solução propõe um gerador automático de regras e blacklists para o SDI, com base em extração de conhecimento OSINT recolhido por uma plataforma de threat intelligence, bem como a sua completude e integração no SDI de forma automática.
- Desenvolver um gerador de regras e blacklists para SDI, com base em informação OSINT recolhida, extraída e agregada por uma plataforma de threat intelligence.
- Implementação da solução proposta num cenário real, na Universidade de Lisboa. Para a implementação, numa primeira instância, é necessário conhecer a rede da universidade, nomeadamente os equipamentos de interligação de redes de computadores, ferramentas de monitorização, registo de eventos e identificação de tráfego na rede.

1.3 Contribuições

As contribuições desta dissertação são as seguintes:

- Um estudo sobre botnets, por forma a caracterizá-los e identifica-los em tráfego de rede.
- Uma arquitetura para a extração de conhecimento OSINT e criação de eventos pela agregação do conhecimento extraído.
- Um gerador automático de regras e blacklists para SDI, com base nos eventos criados.
- Uma implementação automatizada da solução de extração de OSINT, geração de regras e blacklists, e integração no SDI.
- Uma avaliação experimental da solução implementada em ambiente real, na UL.

1.4 Publicações

O trabalho realizado nesta dissertação permitiu a publicação de um artigo completo (12 páginas) nas atas do 9º Simpósio de Informática - INForum 2017, na sessão de *Segurança de Sistemas de Computadores e Comunicações* [17].

1.5 Organização do documento

Este documento encontra-se dividido da seguinte forma:

O capítulo 2 retrata as botnets descrevendo a sua organização, estrutura, comunicação, caraterísticas e os principais ataques que levam a cabo. São também retratados os SDI's através das suas formas de funcionamento, e mecanismos de deteção, de um levantamento dos SDI's mais utilizados pelas organizações, bem como uma apresentação do OSINT. O capítulo também apresenta o trabalho relacionado sobre a temática e finaliza com algumas considerações.

O capítulo 3 apresenta uma visão geral da arquitetura proposta e a forma como está organizada, descrevendo em traços gerais os componentes que a compõe.

O capítulo 4 descreve detalhadamente a implementação da arquitetura proposta. O capítulo subdivide-se pelas suas componentes, apresentando o seu funcionamento, interação e toda a sua estrutura.

O capítulo 5 retrata o ambiente, set-up e performance utilizado na avaliação da solução implementada no capítulo 4. É dada uma visão da rede analisada e quais os incidentes detetados.

Para finalizar, o capítulo 6 apresenta as conclusões do trabalho realizado nesta dissertação e o trabalho futuro advindo dela.

Capítulo 2

Contexto e Trabalho Relacionado

Este capítulo apresenta as características das botnets, SDIs, OSINT e o trabalho relacionado. O capítulo descreve a forma como as botnets se organizam, estruturam, comunicam, quais as suas caraterísticas e os principais ataques que levam a cabo. É também dada uma visão dos indicadores de deteção de botnets na rede. De seguida, são apresentadas as soluções de SDIs mais utilizadas na deteção de ameaças, bem como o seu funcionamento na deteção de botnets. Em terceiro lugar, é apresentado o papel de OSINT na comunidade cibernáutica e como é que está a ser utilizado. Por fim o capítulo apresenta o trabalho relacionado e uma reflexão sobre este na área de deteção de anomalias

2.1 Botnets

2.1.1 Ciclo de vida de um bot

Como já referido anteriormente, um bot é um dispositivo (ex: computador ou smartphone) controlado por um ciber-criminoso com o objetivo de realizar ações ilícitas. No ciclo de vida de um bot são aplicadas uma ou diversas técnicas com o objetivo de explorar uma falha humana ou computacional que permita ganhar acesso à vitima. Estas técnicas podem ser automáticas, ou em casos mais extremos, manuais. As técnicas automáticas têm por norma infetar o maior número de computadores sem grande esforço, e passam geralmente por envio de e-mails maliciosos, instalação de software malicioso, exploração de vulnerabilidades conhecidas, entre outras [18] [19] [13]. As técnicas manuais são iguais às automáticas, mas com um grau de perigosidade bastante mais elevado porque são direcionadas e com um grau de sucesso bastante maior. Um exemplo disso são os emails de *Spearfishing SPAM*, os quais são caraterizados por serem direcionados e cuidadosamente elaborados para dissuadirem o seu leitor, e têm um grau de sucesso elevadíssimo. [20].

O ciclo de vida de um bot contem duas fases principais: *comunicação* e *ataque*.

Comunicação

Para controlar os seus bots, o botmaster utiliza diversas técnicas dependendo da forma como se encontra estruturada a sua rede de bots. A base da infraestrutura de uma rede de bots denomina-se de centro de *Comando e Controle* (Command-and-Control, C&C). É por esta base que o botmaster controla os bots e envia comandos/ordens a eles, ou seja, por onde comunicam [19] [21]. Durante esta fase os bots ligam-se periodicamente ao C&C para receberem diversos tipos de comandos, desde ataques, recolhas de informação e até updates [18]. Esta estrutura permite que o botmaster mantenha um rasto dos seus bots e os controle de forma organizada.

Para comunicar, os bots utilizam diversos tipos de protocolo bem definidos e utilizados por serviços comuns, dissimulando-se assim entre o tráfego benigno.

Alguns destes protocolos são:

- *IRC* – Este é o protocolo mais utilizado devido à sua relação histórica nos grupos de chat, sendo estes problemas de segurança a principal motivação para se descontinuar a sua utilização [19]. Desta forma é fácil enviar mensagens um-para-um ou um-para-todos facilitando a gestão destas redes. Contudo, é fácil bloquear este protocolo nos dispositivos de segurança [18].
- *P2P* – Geralmente o protocolo eleito para estruturar uma botnet descentralizada (ver Seção 2.1.2), isto é, sem C&C [16]. É um protocolo complexo e de difícil deteção devido à sua estrutura e por poder usar criptografia simétrica ou assimétrica [13] [22]. Este protocolo é muitas vezes permitido nos dispositivos de segurança por ser comum entre diversos serviços.
- *HTTP* – O protocolo mais popular e mais utilizado na Internet, por essa razão é também muitas vezes a base de comunicação destas redes maliciosas. A vantagem do seu uso está na impossibilidade prática do bloqueio deste protocolo nos dispositivos de segurança, pois tornaria impossível visitar sites na Internet [18]. Este protocolo dificulta ainda mais a sua deteção devido à sua facilidade em mascarar-se como tráfego benigno [18].
- *DNS* – Com o objetivo de aceder ao servidor C&C, os bots fazem pedidos DNS para o identificarem. Este método possibilita a fácil realocação do C&C entre diversos domínios mantendo o bot sempre a par da sua localização. Contudo, este método facilita a sua deteção pois gera tráfego DNS anómalo na rede, denunciando a sua presença [13].

Ataque

Esta fase é o fruto de todo o trabalho do *botmaster* e demonstra a verdadeira finalidade dos bots, ou seja, a realização de ações maliciosas. Podem ser diversos os tipos de ações maliciosas desencadeadas pelo botmaster, sendo as mais frequentes as seguintes:

- *DDoS (Distributed Denial of Service)* – Este tipo de ataque tem como objetivo reduzir parcial ou completamente a disponibilidade de um serviço, e se bem-sucedido pode ter custos elevadíssimos para empresas financeiras e de retalho [18] [23]. Estima-se que 49% dos ataques tenham uma duração entre 6 e 24 horas e tenham um custo estimado de 37.500€ por hora, sendo que em média, o prejuízo causado por este tipo de ataques é de aproximadamente 47.000€ [23].
- *SPAM* – O envio de mensagem em massa não solicitadas bem como outro tipo de atividades de SPAM, provêm entre 80% e 85% de botnets [13]. Desta forma os ciber-criminosos infetam outros computadores, que por sua vez aumentam a quantidade de SPAM gerado. Um bot consegue enviar por segundo, em média, mais de 3 emails ou mensagens falsas [18].
- *Roubo de Informação Pessoal* – Com acesso aos seus bots, o botmaster pode recolher diverso tipo de informação. Este método serve geralmente para roubar credenciais pessoais de acesso a recursos financeiros com o intuito de roubar capital ou outros recursos relevantes. Também pode ser feita recolha de informação de websites [21].

Existem muitos outros tipos de ações que podem caraterizar um bot, como por exemplo os bots que personificam browsers legítimos, motores de busca, monitorização de serviços, etc.

2.1.2 Arquitetura

As botnets podem assumir quatro tipos de arquiteturas: *centralizada*, *descentralizada*, *híbrida* e *aleatória*. Cada arquitetura tem características próprias que a distingue das restantes arquiteturas.

Arquitetura Centralizada

Esta topologia é caracterizada pelo C&C ser uma entidade centralizada onde é efetuada a troca de mensagens entre os bots e o botmaster. Esta entidade é escolhida pelo botmaster e pode ser uma das suas vítimas ou um servidor dedicado. Após uma máquina ser infetada, esta liga-se ao C&C e aguarda comandos. Quando um comando é enviado pelo botmaster através do C&C, os bots que se encontram ligados ao C&C recebem o comando e o executam-no [18]. Esta arquitetura é de fácil gestão para o botmaster, no entanto, ao desmantelar-se o C&C, desmantela-se toda a rede [13]. A Figura 1 ilustra esta arquitetura.

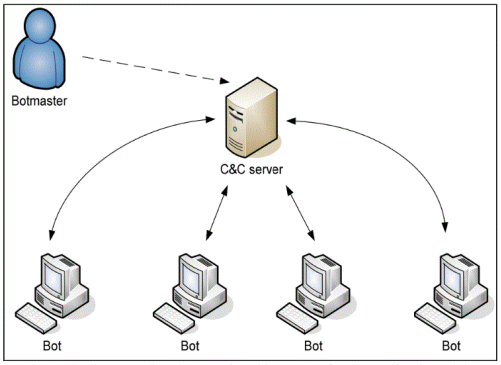

Figura 1 - Arquitetura Centralizada [18]

Arquitetura Descentralizada

Nesta arquitetura não existe C&C. O botmaster para enviar um comando conecta-se a qualquer bot pertencente à botnet, como ilustra a Figura 2 [18]. O comando é depois replicado (ou não) para os restantes bots pertencentes à botnet. Desta forma, torna-se mais complicado gerir a botnet, no entanto, como não existe uma entidade centralizada, torna-se mais complicado desmantela-la [13].

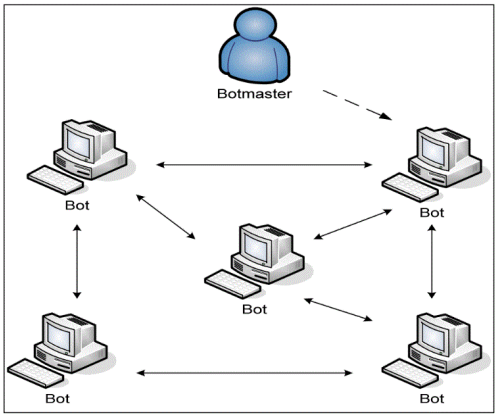

Figura 2 - Arquitetura Descentralizada [18]

Arquitetura Híbrida

Este é a arquitetura mais complexa de ser gerida, no entanto é a mais modular, organizada, expansível e de difícil deteção. É concebida com o melhor das arquiteturas centralizada e descentralizada, sendo complexa e tendo a vantagem da utilização de diversos protocolos. Esta é caracterizada pelo seu C&C ser constituído por vários bots servidores que se encarregam de funcionar como ponto de comunicação entre bots e botmaster [13] [18]. Esta arquitetura encontra-se ilustrada na Figura 3.

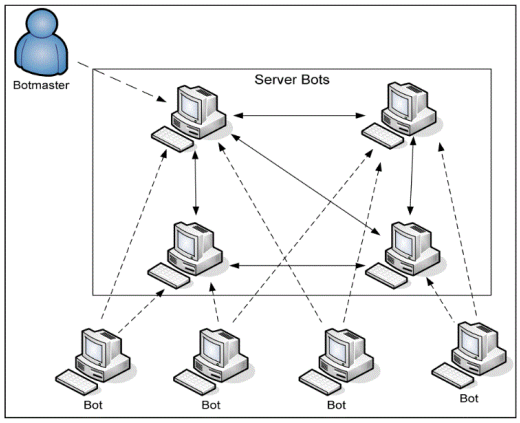

Figura 3 - Arquitetura Híbrida [18]

Arquitetura Aleatória

Uma arquitetura considerada teórica na aplicação de botnets. Baseada no princípio de que o bot não contata ativamente o botmaster ou outros bots. Nesta arquitetura o botmaster é quem tenta contatar os bots da rede e, caso os encontre, envia-lhes comandos. Desta forma um bot torna-se mais difícil de detetar tornando-se mais resiliente por não haver comunicações constantes com o botmaster. Contudo, existe uma perda na escalabilidade e na arquitetura. Atualmente não existe conhecimento de bots que utilizem esta arquitetura [13].

2.1.3 Comportamento

Apesar da complexidade do problema que o torna difícil de solucionar, a verdade é que o tráfego é detetável. Existem muitas variantes na sua deteção, desde o conteúdo do tráfego gerado pelos bots, à inspeção de máquinas com padrões suspeitos em determinados tipos de comunicação.

As botnets possuem determinadas características, que a partir das quais torna possível a sua deteção. Estas caraterísticas podem diferir consoante a estrutura ou

protocolo de comunicação entre bots e o centro de C&C. Algumas das características que denunciam os bots assentam na arquitetura da botnet, onde eles se inserem e no protocolo em uso. Estas são apresentadas de seguida:

- *Informação do cabeçalho HTTP* – O cabeçalho HTTP gerado pelos bots é muitas vezes incompleto ou errado, denunciando este tipo de comunicações [18] [24].
- *Tempo e tamanho de resposta* – Um bot responde ao botmaster a uma velocidade inalcançável por um humano. A isto alia-se o facto de que, respondem em pequenas quantidades para passarem despercebidos na rede, gerando pouco tráfego [18].
- *Execução de comandos* – Após receberem comandos do botmaster, os bots podem ter a necessidade de executar aplicações [25].
- *Comportamento periódico* – Dependendo da arquitetura da botnet, periodicamente os bots comunicam com o centro de C&C para receberem comandos, atualizações ou mensagens a divulgar o seu estado [13].
- *Densidade da comunicação* – Durante um ataque, por exemplo de *DDoS*, ao contrário da comunicação entre o C&C, existe um pico de tentativas de ligação ao serviço alvo, resultando num volume de tráfego anormal [18]. A Figura 4 ilustra um desses picos na parte direita.

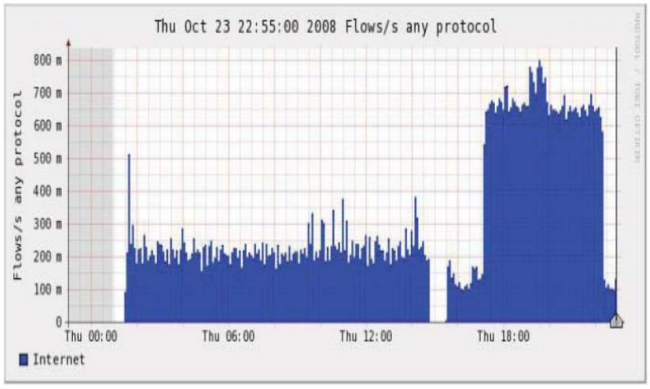

Figura 4 - Fluxo de uma botnet durante um ataque [18]

A presença destas características permite entender os pontos fundamentais do problema e perceber o porquê do seu poder destrutivo e da sua difícil resolução. A estrutura das redes de bots pouca ou nenhuma evolução sofreu na última década, no entanto, a evolução dos protocolos standard foi aproveitada para aprimorar as suas técnicas de dissimulação.

2.2 Sistemas de Deteção de Intrusão

O primeiro SDI denominava-se *Intrusion Detection Expert System* (IDES) e foi desenhado na década de 80 por Dorothy Denning e Peter Neumann, alguns anos após James P. Anderson ter publicado um trabalho sobre como utilizar a informação de eventos registados por serviços computacionais para deteção de acesos indevidos [12]. Este trabalho alertou a comunidade sobre a necessidade de haver formas de detetar utilizações indevidas em mainframes. Essa necessidade evoluiu conjuntamente com as tecnologias até aos dias de hoje [12].

Estes sistemas têm como finalidade monitorizar e analisar eventos que ocorrem num computador ou numa rede, verificando se o comportamento dos seus utilizadores entra em conflito com o correto funcionamento do sistema [26].

Apesar da importância deste tipo de sistemas na deteção de bots, a sua evolução não acompanhou as necessidades do mundo atual [14]. O défice de desenvolvimento destes sistemas deve-se, na sua grande maioria, devido à necessidade de cooperação e troca de informação entre diferentes entidades. Isto acontece porque muita dessa informação é considerada segredo de negócio ou informação privilegiada. Havendo esta dificuldade, os investigadores estão limitados a pequenas amostras de informação, colocando em causa a qualidade da solução em ambiente de produção [14].

2.2.1 Mecanismos de deteção

Para detetar comportamentos anómalos existem duas categorias de técnicas de deteção, através de honeynets (redes intencionalmente desprotegidas) e de SDIs. Enquanto as honeynets se focam, na sua grande maioria, na obtenção de informação de bots, os SDI utilizam informação para detetar bots em atividade [13].

Os mecanismos de deteção dos SDIs encontram-se divididos em duas subcategorias, nomeadamente *assinaturas* e *comportamento* [13] [12].

Assinaturas

Com recurso a bases de dados contendo assinaturas de ameaças conhecidas, esta categoria de SDI's faz a análise de pacotes, procurando uma possível assinatura conhecida que denuncie a presença de uma intrusão.

Apesar da trivial implementação de sistemas deste tipo, esta solução apenas permite detetar bots de assinaturas conhecidas, que constem na base de dados, e cujo comportamento não divirja do expectável.

A grande vantagem deste método é o baixo número de falsos positivos, i.e., falsos alarmes despoletados. No entanto, as intrusões desconhecidas podem originar falsos negativos [13] [12].

Comportamento

Esta é a categoria mais investigada na área de deteção de botnets. Tendo como primitiva o total conhecimento do correto funcionamento do sistema em que se insere, este tipo de SDIs consegue detetar padrões considerados anormais e gerar alarmes relativos a esses desvios comportamentais. Este tipo de sistemas são complexos e envolvem um estudo extensivo e constante do sistema que vai analisar.

Contrariamente aos SDIs baseados em assinaturas, esta categoria tem um número muito baixo de falsos negativos conseguindo detetar intrusões com comportamentos desconhecidos e divergentes, como por exemplo, exploração de vulnerabilidades. Contudo esta solução dá aso a um número bastante elevado de falsos positivos.

2.2.2 Formas de funcionamento

Tendo em conta o funcionamento dos SDI, estes podem ser aplicados em dois lugares distintos para monitorizar o comportamento do sistema, sendo estes o *network-based* e o *host-based* [13] [27].

Network-based

Como o próprio nome indica, este tipo de SDI's é integrado ao nível de rede do sistema em que se insere. Para detetar possíveis intrusões é recolhida informação e inspecionado o tráfego que o sistema gera. Através de informações retiradas do cabeçalho do pacote ou até à própria payload, o SDI deteta se existem comportamentos anómalos e que representem uma ameaça ao bom funcionamento do sistema.

Uma grande vantagem deste tipo de sistemas é a sua autonomia física, quer isto dizer que, caso alguma máquina da rede passe a ser controlada por uma entidade criminosa, esta não consegue desativar o SDI. Outra grande vantagem desta autonomia é

que, para além de não afetar a performance do sistema monitorizado, ainda consegue detetar ataques a múltiplos alvos do sistema [27].

Host-based

Este tipo de SDI faz parte integral do sistema que monitoriza. Esta aproximação permite recolher informação detalhada do sistema, como por exemplo, a duração das sessões, comandos, ficheiros acedidos e outras informações consideradas relevantes. Isto torna-se especialmente importante quando, por exemplo, o tráfego gerado pelas botnets é cifrado, pois permite interpretar dados que, através de monitorização ao nível da rede, seria impossível.

Contudo, esta topologia é intrusiva e a análise pode ser tendenciosa a partir do momento em que uma entidade criminosa toma conta do sistema monitorizado. Como o próprio sistema, para além de realizar as tarefas propostas, tem de realizar tarefas de análise, por vezes ocorrem decréscimos significativos da performance [27].

2.2.3 SDIs mais utilizados

Atualmente não existe um grande número de SDIs no mercado, sendo que os existentes são, na grande maioria, gratuitos e opensource. Tal como descrito na secção anterior, os SDIs disponíveis estão subdivididos em network-based e host-based.

Os network-based que merecem uma maior atenção são:

Snort

Criado por Martin Roesch em 1998, este SDI conta com mais de 4 milhões de downloads, sendo então considerado o mais implementado no mundo. Em 2009 foi considerado um dos melhores softwares opensource de todos os tempos, tendo sido adquirido pela *CISCO* 4 anos depois [28] [29] [30].

O Snort é escrito na linguagem de programação C e é capaz de analisar tráfego, correlacionar pacotes e ainda funcionar como prevenção de intrusões, através de regras. As regras geram alertas que, contrariamente às assinaturas, são utilizadas também para detetar diversas vulnerabilidades, trazendo a vantagem de detetar problemas ainda não conhecidos [28].

A sua arquitetura modular permite que exista um balanço entre a qualidade da análise do tráfego e performance. Para detetar uma intrusão, o Snort recebe dois tipos de informação, regras e IP blacklisted. Enquanto as regras permitem encontrar padrões suspeitos no tráfego, os IP blacklisted identificam tráfego provenientes de máquinas com má reputação e maliciosas.

O motor de alertas do Snort processa as regras, as quais contêm informação sobre um determinado evento e simboliza um padrão que identifica, com grande probabilidade, uma intrusão. Este motor tem uma grande capacidade de deteção, no entanto, tem um custo de processamento grande, ou seja, quanto maior for a lista de regras, menor é a performance do sistema.

Performance

Dois dos principais fatores que diminuem a performance do Snort prendem-se com o crescente número de regras que o SDI tem de processar ou quando as regras de alerta possuem uma elevada quantidade de IPs de má reputação. Para resolver esta situação, o Snort contém um preprocessador baseado em reputação (verifica se um IP se encontra na blacklist). Este preprocessador entra em ação logo após ter ocorrido a descodificação do pacote, comparando-o com os IPs constantes na blacklist. Caso o IP se encontre nessa lista, é gerado um alerta e não é feita nenhuma análise por parte do motor de alertas, desafogando assim a quantidade de processamento do Snort [31].

Outro problema que é facilmente resolvido no Snort é a gestão dos IPs que se pretendem monitorizar. Para este efeito existe o Barkley Packet Filtering, um mecanismo que utiliza regras para filtrar tráfego antes que seja processado pelo Snort, permitindo também assim um ganho de performance bem como flexibilidade de gestão.

Regras

As regras são a linguagem e a funcionalidade core do Snort. É através delas que definimos determinados padrões no tráfego que denunciam uma possível intrusão. As regras são compostas da seguinte forma:

```
[action][protocol][sourceIP][sourceport] ->
[destIP][destport] ([Rule options])
```

- *Action* – Designa o que vai acontecer se a regra for acionada, sendo a ação mais utilizada e relevante para a solução o alert. Esta ação permite gerar uma

mensagem que identifica que esta regra foi disparada e guarda o pacote que a fez disparar para uma futura análise.

- *Protocol* – Este campo serve para selecionar o protocolo sobre o qual incide a regra, podendo ser TCP, UDP, ICMP e IP.
- *SourceIP* – Especifica o endereço IP que originou o pacote.
- *Sourceport* – Porto pelo qual o pacote foi enviado.
- *DestIP* – Endereço IP a quem o pacote se destina.
- *Destport* – Porto a que o pacote vai ser entregue.
- *Rule options* – O campo rule options é o coração da regra. É sobre diversas opções que se vai analisar a payload do pacote para detetar padrões que indiquem uma tentativa de intrusão ou outro comportamento desejado.

Para detetar uma intrusão é necessário prover o Snort com informação sobre o comportamento dessa ameaça através de uma regra com os campos descritos anteriormente. Por exemplo, a seguinte regra:

```
alert tcp 192.168.4.28 any ->
10.100.55.6 80 (msg:”Acesso ao site!”; sid:1000000])
```

Com a identificação (*sid*) número *1000000,* vai gerar um alerta com a mensagem *“Acesso ao site!”* sempre que o IP *192.168.4.28* aceder através de qualquer porto (*any*), ao porto *80* (*HTTP*) da máquina com o IP *10.100.55.6*.

Estas características tornam o Snort um sistema dependente de uma fonte de conhecimento humana (ou não). Estas regras são habitualmente produzidas internamente nas instituições pela equipa de cibersegurança ou por empresas, e posteriormente vendidas. Contudo algumas empresas disponibilizam regras gratuitamente na Internet mas são limitadas e servem maioritariamente para aliciar a comprar a solução profissional. Estas empresas não disponibilizam, por exemplo, regras relativas a Phishing.

Suricata

Sendo opensource e tendo como proprietário a *Open Information Security Foundation* (OISF), o Suricata é dos mais bem-sucedidos SDI.

Com um motor capaz de analisar tráfego, prevenir intrusões e processar pacotes offline, o Suricata é também dotado de um ambiente gráfico que permite monitorizar facilmente a segurança da rede.

Os seus mecanismos de deteção são baseados em regras e em assinaturas permitindo reconhecer intrusões conhecidas e não conhecidas [32].

Bro

Contando com mais de 20 anos de investigação e com apoios como o *mozzilla* e da *National Science Foundation* (NSF), o Bro é um SDI a ter em conta.

Adaptável, eficiente, flexível e opensource, o Bro ainda consegue fazer análise forense. Apesar de ser escrito em C++, este SDI apenas funciona em sistemas Linux.

O Bro faz análise de tráfego com um foco particular na monitorização de segurança em escala, através de assinaturas e comportamento.

Para além das características descritas, este SDI é muito utilizado por investigadores para analisar propriedades de redes e prototipar novos tipos de análises, sendo este o seu ponto forte [33] [34].

Kismet

O Kismet é um SDI para redes sem fios. Capaz de analisar tráfego e detetar intrusões, também consegue identificar redes e colecionar pacotes para descobrir redes escondidas.

Com uma arquitetura que suporta plugins, o Kismet consegue descodificar protocolos que não sejam integradas na norma 802.11.

Este SDI permite o desenvolvimento de plugins para estender a capacidade de análise, adicionar alertas, adicionar novos pontos de captura e melhorar a interface gráfica [35].

Os SDI host-based mais importantes e atualizados são:

OSSEC HIDS

O OSSEC permite monitorizar ativamente toda a atividade de sistemas UNIX através da integridade de ficheiros, logs, privilégios e processos.

Através de logs de alertas e até emails, este SDI permite tomar ações rápidas quando existe a ocorrência de uma intrusão. Este sistema permite ainda uma fácil integração com ferramentas de gestão e correlação de eventos de segurança.

Sendo gratuito e opensource, é possível modificar o sistema de modo a adicionar novas funcionalidades. Este SDI contém ainda um vasto leque de configurações facilitando a integração e tornando-o ajustado à estrutura a monitorizar [36].

Samhain

Para além das tarefas comuns dos SDI host-based como a monitorização de logs e verificação da integridade de ficheiros, o Samhain consegue detetar rootkit's, executáveis maliciosos, processos escondidos e monitorizar portos.

Desenhado para monitorizar múltiplas máquinas, este SDI consegue lidar com sistemas com um elevado grau de heterogeneidade, disponibilizando uma unidade centralizada de manutenção e logging.

O Samhain é opensource, compatível com sistemas POSIX e contém um ambiente baseado em web chamado Beltane para facilitar a monitorização [37].

2.3 OSINT

OSINT, ou Open Source Intelligence, é o termo dado a informação recolhida e tratada a partir de OSINFs (Open Source Information) [38]. As OSINFs são dados públicos em bruto, recolhidos de diversas fontes de informação como por exemplo um site, um tweet, uma entrevista ou até mesmo uma fotografia [39] [38]. Após essa recolha, a informação é tratada, organizada, agregada e classificada dando origem a OSINTs [38]. Contudo, após as OSINFs serem transformadas em OSINTs, a informação neles contidas pode passar a ser considerada confidencial [38]. Entre todas as temáticas informativas encontradas nas OSINT/OSINFs, estão os eventos de segurança relativo a ameaças informáticas.

A grande vantagem deste tipo de informação é a velocidade a que pode ser acedida permitindo uma rápida e precisa tomada de decisão em tempo real sobre um determinado evento de segurança, acrescido de que, por não se tratar de informação confidencial, poder ser partilhada entre diversas organizações [38]. Este tipo de informação é mais barata e pode ter mais qualidade do que a informação confidencial, pode ser encontrada em diversos sítios e o preço do software necessário para a sua recolha e análise é quase nulo.

Contudo, a enorme quantidade de informação disponível pode tornar-se uma desvantagem na medida em que a sua análise se torna mais complexa e difícil de

processar. Não existe também um método de classificação do valor da informação disponível, sendo necessário pessoas especializadas para realizar este trabalho.

Este tipo de abordagem na recolha e utilização da informação pode ter consequências na vida real como foi o caso de um grupo de turistas ingleses terem sido exportados dos estados unidos por terem escrito no Twitter que iam “rebentar” com a américa, não no contexto terrorista, mas festivo [40]. Este tipo de situações sugere que a informação recolhida e tratada deve ser utilizada de forma responsável e segura [39].

Os negócios que assentam sobre este tipo de principio estão em crescimento [39] sendo visível na quantidade de feeds sobre eventos de segurança informática existentes. Alguns destes feeds provêm de organizações, como o Phistank [41] da OpenDNS, e são disponibilizados gratuitamente, ou de empresas privadas com fins lucrativos como é o caso da CINScore [42].

Posto isto, este tipo de informação é relevante na medida em que pode ser utilizada para ficar a par de eventos de segurança e de ameaças que podem servir para tornar organizações mais seguras, e o ciberespaço de uma forma geral.

2.4 Trabalho Relacionado

Zeng et al. [43] apresentaram uma arquitetura híbrida (baseada em rede e em host) para a deteção de botnets que tinha como principal caraterística a interseção de dados provenientes da rede e dos computadores. Numa primeira fase, a arquitetura, através da análise de tráfego, identifica computadores suspeitos, cujo comportamento seja muito diferente dos restantes. Numa segunda fase, a arquitetura através de análise baseada em host valida se esses computadores são de facto suspeitos. Este método permite uma avaliação mais precisa, pois para além de analisar o comum comportamento de coordenação intrínseco às botnets, também realiza uma análise ao seu comportamento como um só.

Sperotto et al. [44] propuseram um SDI, que por questões de performance e em redes de alta velocidade (ex., 10Gbps), analisa o fluxo do tráfego na rede em vez de analisar o pacote todo. Contudo, o SDI só consegue detetar ataques de negação de serviço, scans, worms e botnets, devido ao facto de somente analisar os cabeçalhos dos pacotes. Contrariamente à solução proposta neste artigo, este tipo de SDI não consegue detetar ataques de Phishing, por exemplo, uma vez que é necessário analisar o conteúdo do pacote [44].

Um survey relativo à deteção de botnets [13] mostra que, apesar do avanço em alguns métodos de deteção, poucos foram bem-sucedidos e que a técnica mais usada para na deteção é a baseada em anomalias. Contudo, continua a existir uma grande dificuldade no desmantelamento de botnets porque os botmasters continuam a evoluir técnicas de mascaramento. Aviv et al. [14] propõe uma cooperação de diversas entidades para desenvolvimento de mecanismos para deteção de botnets, uma vez que estas entidades possuem informação confidencial.

Para além do funcionamento standard dos SDI, existem SDI específicos para determinado tipo de ataque e que utilizam técnicas que não as standard. O FeatureSmith é um sistema de deteção de malware em aplicações android cujas funcionalidades assentam em mineração de dados. Para a aplicação de mineração de dados o sistema coletou dados sobre malware em android de 1068 artigos científicos do google scholar, onde através do relacionamento semântico com o comportamento de malware e do seu mapeamento para funcionalidades o sistema conseguiu uma taxa de deteção de 92.5% de positivos verdadeiros, com uma taxa de 1% de falsos positivos [45]. Ao contrário do FeatureSmith, a solução apresentada baseia-se em deteção de padrões conhecidos de intrusões, recolhidos de eventos OSINT e processados para criação de regras e identificação de endereços IP maliciosos, que serão usados pelo SDI.

O MISP [46] é uma plataforma colaborativa de partilha de informação de ameaças de segurança. A sua arquitetura baseia-se em organização e comunidade. Estes dois grupos partilham informação de forma simples e controlada com o objetivo de alertar as entidades pertencentes à organização ou a determinada comunidade.

O BotSniffer [47] é um sistema desenvolvido por Guofei et al. e tem como principal objetivo detetar C&C de botnets na rede através da observação de tráfego. Esse sistema é um SDI network-based que assenta na deteção através de anomalias. Recorrendo a meios estatísticos e de correlação de tráfego espácio-temporal, os autores deste sistema criaram uma heurística que permite ao sistema detetar atividades deste tipo de ameaças. Testes em ambiente real demonstraram que esta solução era capaz de detetar botnets com um baixo número de falsos positivos. Este estudo permitiu também perceber que as atividades das botnets que utilizam o mesmo programa, fazem comunicações muito parecidas entre si e durante os ataques, demonstrando similaridade de comportamento.

Um estudo realizado por Gregory et al. [48] apresenta um método de deteção host-based para detetar e diferenciar botnets de acordo com as suas comunicações com o C&C, categorizando-os. O estudo teve um contributo importante pois foi desenvolvido um

método através do qual foi possível identificar comunicações de botnets através de IRC sem inspecionar a payload dos pacotes da comunicação e com um baixo número de falsos positivos.

Rishi [49] é um detetor de bots que utilizam canais IRC para comunicar. Este programa destaca-se por fazer a deteção através do nome que estes bots utilizam na comunicação por esses canais. Para perceber se estes nomes são utilizados por bots, são geradas expressões regulares que permitem identificar se um nome foi gerado automaticamente para ser utilizado numa botnet. Como prova de funcionamento, o Rishi foi implementado em Python demonstrando resultados interessantes, contudo chegou-se à conclusão que não seria possível utilizar o sistema de forma automatizada e autónoma sem um largo número de falsos positivos.

Um estudo realizado por investigadores da Microsoft permitiu desenvolver uma framework denominada de AutoRE [50], capaz de gerar assinaturas URL que permitem detetar botnets que geram emails de SPAM. Estas assinaturas utilizam expressões regulares para detetar padrões nos URLs, situação que até a altura era feita por humanos. O estudo conclui que a ferramenta desenvolvida tem potencial para ser utilizada em tempo real para parar uma grande quantidade de campanhas de SPAM. Os resultados demonstram uma quantidade insignificante de falsos positivos. Contudo a evolução das botnets levou à utilização de URL polimórficos, obrigando a que este tipo de metodologia tenha de evoluir para acompanhar esta tendência.

Peter et al. [51] desenvolveram um sistema que permite detetar botnets sem grande conhecimento relativo aos bots e ao C&C. Através da análise de tráfego de rede com o SDI Bro, é possível detetar este tipo de ameaças. Utilizando como input alguns binários de bots e algumas das suas comunicações na rede, são gerados modelos que servem como conhecimento ao SDI para detetar diferentes tipos de famílias de bots. Este sistema baseia-se principalmente nas características do envio e receção das comunicações estabelecidas entre os bots e o C&C para detetar ações maliciosas e identificar as ameaças com um número muito baixo de falsos positivos.

O retrato do estado da arte demonstra persistência e trabalho continuo no combate ao uso e apropriação ilícita de recursos computacionais. Com base em técnicas de deteção de intrusões network-based e host-based, têm sido acrescentadas caraterísticas e funcionalidades que permitiram um aumento de verdadeiros positivos e uma redução de falsos positivos.

A proposta de solução enquadra-se em algumas das caraterísticas encontradas no trabalho relacionado. Aliada a algumas técnicas presentes como o uso de um SDI network-based e recolha de informação, é uma solução que permite detetar ameaças na rede através de geração de conhecimento com informação recolhida em eventos de segurança, contribuindo desta forma com uma arquitetura de deteção de intrusões inovadora.

2.5 Considerações Finais

A capacidade, constante evolução e evasão dos criminosos na aplicação das suas técnicas no roubo de recursos e na sua gestão, demonstra que este problema não tem uma solução trivial. Diversas técnicas têm sido aplicadas com o objetivo de prevenir ou mitigar este problema, contudo, os cibercriminosos têm evoluído para contrariar essas técnicas, tal como é visível no trabalho relacionado, sendo que das ferramentas mais bem-sucedidas na deteção deste tipo de problemas são os SDI host e network-based.

Com recurso a SDIs, os investigadores têm evoluído estas arquiteturas com o objetivo de chegar a uma solução que permita minimizar ou resolver este problema. Algumas abordagens permitem detetar as ações maliciosas praticadas pelos botmasters, através da deteção de padrões no tráfego ou no comportamento gerados pelos seus bots.

Este estudo permitiu propor uma arquitetura que, através da recolha de informação de OSINTs, consegue detetar ameaças e manter-se atualizado de forma automática e sem intervenção humana. Com recurso a um SDI network-based, aliado a uma plataforma que permite recolher e tratar informação de eventos de segurança constantes em OSINTs, é gerado conhecimento diariamente, o qual é integrado automaticamente no SDI, revelando-se assim uma solução prática no combate a ameaças.

Capítulo 3

Geração de Regras e Blacklist para SDI usando OSINT

Este capítulo retrata uma solução modular que permite detetar ameaças tendo como base os conceitos apresentados anteriormente. É descrita a base que permitiu chegar à solução apresentada, é dada uma visão da composição da arquitetura e como cada um dos seus componentes se interliga e comunica.

3.1 Visão geral da arquitetura

A arquitetura apresentada tem como objetivo detetar bots e outras ameaças conhecidas por atacarem organizações ou descobertos por aficionados e que foram publicadas na Internet.

Em seguimento do estudo apresentado no capítulo anterior, foi desenhada uma arquitetura com uma forte ligação com a cooperação na partilha de eventos de segurança tal como o sugerido por Adam Aviv et al [14]. Para tal, são utilizados como fonte de informação feeds OSINT, para serem geradas regras como forma de detetar o comportamento de bots e outras ameaças na rede. Para a recolha e tratamento da informação provinda desses feeds, é utilizada uma plataforma threat intelligence. Posteriormente essa informação servirá para gerar essas regras irão permitir detetar o tráfego gerado por estas ameaças.

São utilizadas as regras geradas como conhecimento para um SDI network-based para a deteção da presença de ameaças na rede. Como forma de manter o SDI atualizado diariamente sobre as ameaças existentes, é realizado um processo de atualização de forma automática e autónoma através de um gestor de regras e blacklists.

A arquitetura da solução proposta é composta por três componentes: *recolha de informação*, *geração de conhecimento* e *deteção de incidentes*. A primeira recolhe e agrega informação de eventos de segurança de feeds OSINT, enquanto a segunda gera conhecimento para SDI, sob a forma de regras e blacklists, com base nesta informação, e a terceira deteta incidentes pela aplicação do conhecimento gerado.

As três componentes estão representadas na Figura 5. As duas primeiras componentes são executadas todos os dias, para recolha e processamento de novos eventos de OSINT, e geração de conhecimento para SDI a partir destes. A terceira componente está em contínua execução. Mais detalhadamente as componentes operam da seguinte forma:

Recolha de Informação

A componente de recolha de informação executa os seguintes passos: o *OSINT colector* recolhe os eventos de segurança, que no âmbito da arquitetura são vistos como IOCs (*Indicators of Compromise*) [52], dos diversos feeds de OSINT. Seguidamente o módulo *Threat information extractor* processa os eventos/IOCs, extraindo a informação relevante sobre ameaças e organizando-a num formato que a permite manusear de uma forma otimizada. Seguidamente essa informação é agregada pelo módulo *Threat information aggregator*, para facilitar a sua utilização pelos módulos dos outros componentes.

Geração de Conhecimento

A componente de geração de conhecimento, através do módulo *Event generator* recebe a informação agregada, adiciona-lhe dados relevantes sobre identificação de ataques, criando assim IOAs (*Indicators of Attack*, contem informação que permite identificar ataques em execução) [53] [52]. Desta forma as regras de SDI têm mais informação, são mais otimizadas e geram um número menor de falsos positivos, formata também toda a informação, cria novos eventos de segurança e armazena-os. Por fim, o *Rules & blacklist generator* recebe os IOAs gerados pelo módulo anterior e gera regras conforme o tipo e a causa do evento para o SDI, no formato especificado do SDI, e blacklists com IPs maliciosos.

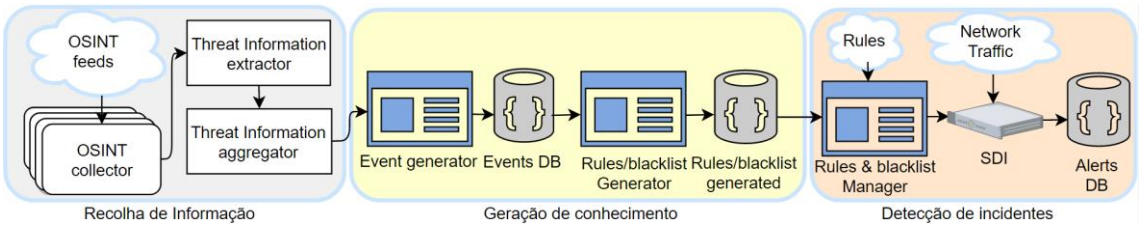

Figura 5 - Arquitetura da solução proposta

Deteção de Incidentes

Por fim este ultimo componente tem como objetivo a de deteção de incidentes. O *Rules & blacklist manager* gere as novas regras e blacklists, vindas da componente geração de conhecimento e do exterior (representado por *Rules* na figura), para de seguida fornecer tal conhecimento ao SDI para detetar incidentes. Os incidentes são depois colocados na Alerts DB para posteriormente serem analisados por uma equipa de segurança.

Capítulo 4

Implementação

A arquitetura proposta foi implementada usando a plataforma de threat intelligence IntelMQ para coletar eventos de segurança de 49 OSINTs de diferentes categorias, extrair e agregar informação destes. Os módulos event generator e o Rules & blacklists generator foram desenvolvidos no âmbito da dissertação para, respetivamente, interagir com o IntelMQ e processar os seus resultados. A implementação foi obtida pelas fases de gestão do conhecimento e deteção de incidentes, como ilustra a Figura 6. A parte esquerda da figura, gestão do conhecimento, implementa as componentes de recolha de informação e geração de conhecimento da arquitetura proposta, enquanto a parte direita da figura implementa a componente de deteção de incidentes. A seção apresenta de seguida os detalhes e funcionamento de cada uma das componentes.

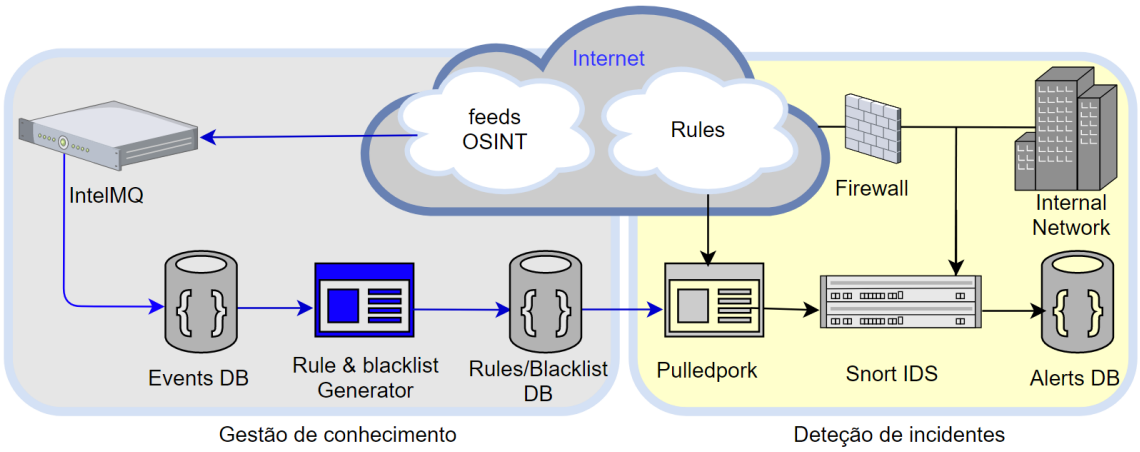

Figura 6 - Implementação da arquitetura proposta, com gestão de conhecimento (esquerda) e deteção de incidentes (direita)

4.1 OSINT feeds

Com o objetivo de reduzir o cibercrime e salvaguardar a integridade intelectual, existem empresas, aficionados, investigadores e instituições (feeds) que disponibilizam informação à comunidade cibernética sobre incidentes de segurança que podem ser utilizados como IOCs, tais como IPs maliciosos, sites de phishing, domínios perigosos, entre outras informações relevantes. Alguns exemplos de feeds, de acordo com a especialidade, são: Phishtank [41] para phishing e CINSScore [42] para reputação de IPs. Esta informação se usada adequadamente e atempadamente pode prevenir a concretização de incidentes perigosos, tais como ataques e roubo de informação. A partir destes feeds, após processamento pelo IntelMQ, serão geradas as regras e blacklists para o Snort. Nesta arquitetura são utilizados 44 repositórios de OSINT gratuitos. A Tabela 1 apresenta a divisão destes por categorias, sendo que alguns se intersectam, o que totaliza os 49 feeds.

Evento OSINT	Nº feeds OSINT
Domínio Malicioso	9
Blacklist IP	16
Phishing ou links perigosos	7
IPs que atacam servidores FTP	1
IPs que atacam servidores de EMAIL	2
IPs que atacam serviços de VoIP	4
IPs que efetuam scans através de SNMP	1
IPs que tentam obter acesso remoto	9
Total	**49**

Tabela 1 - Categorização dos OSINT

4.2 IntelMQ

A ferramenta IntelMQ foi desenhada pelo CERT (Computer Emergency Response Team) europeu com o objetivo de facilitar a recolha e tratamento de dados relativo a ameaças, melhorando desta forma a resposta a incidentes. Esta solução assenta na recolha e processamento de dados de segurança, pastebins, tweets e ficheiros de log através de um protocolo de fila de mensagens [54].

Através de princípios como o KISS (Keep It Simple, Stupid) que valoriza a simplicidade do projeto, descarta toda a possível complexidade, e sendo Opensource, aponta para uma estrutura que permite uma fácil integração a programadores

inexperientes. Com estes princípios em mente, a escrita de bots (atenção para não confundir com bots controlados por entidades criminosas) torna-se trivial. No universo IntelMQ, os bots são blocos de código modulares e independentes na linguagem Python, que comunicam entre si sob forma de mensagens. É utilizado o formato JSON, como ilustra a Figura 7, para a troca de mensagens entre os bots permitindo uma fácil comunicação.

```
array(
      array(
            'phish_id' => 123456,
            'url' => 'http://www.example.com/',
            'phish_detail_url' => 'http://www.phishtank.com/phish_detail.php?phish_id=123456',
            'submission_time' => '2009-06-19T15:15:47+00:00',
            'verified' => 'yes',
            'verification_time' => '2009-06-19T15:37:31+00:00',
            'online' => 'yes',
            'target' => '1st National Example Bank',
            'details' => array(
                        array(
                              'ip_address' => '1.2.3.4',
                              'cidr_block' => '1.2.3.0/24',
                              'announcing_network' => '1234',
                              'rir' => 'arin',
                              'detail_time' => '2006-10-01T02:30:54+00:00'
                             )
                        )
           )
)
```

Figura 7 - Formato de mensagem JSON [25]

Este formato facilita a filtragem da informação na comunicação entre os bots, possibilitando o manuseamento da informação estritamente necessária. Organizado num array, a mensagem disponibiliza um conjunto de informação que pode ou não ser utilizada pelos bots como por exemplo o endereço IP, data de submissão, url, entre outras informações relevantes.

A Figura 8 apresenta a visão geral do funcionamento do sistema, com as diversas componentes que o integram. A definição de bots torna o IntelMQ numa ferramenta a considerar no processamento de feeds de incidentes de segurança. O componente Redis fornece memória cache para facilitar a comunicação entre bots. Escritos em Python, os bots ganham uma dinâmica que permite agilizar o tratamento e troca de informação para comunicar com outras ferramentas.

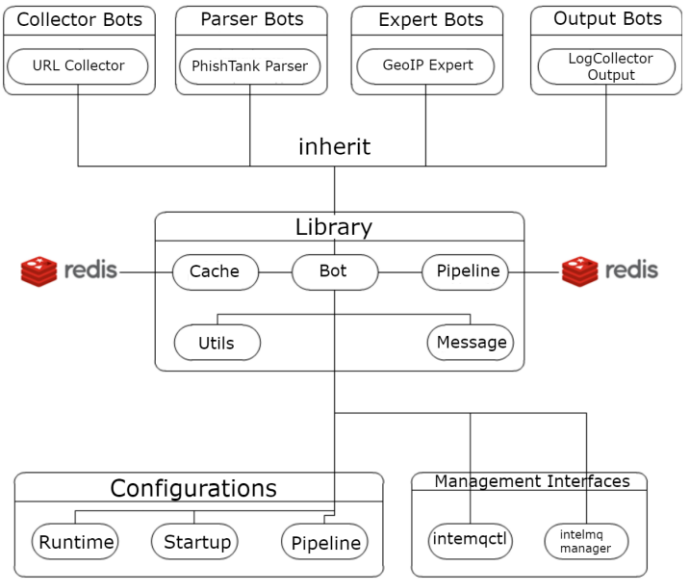

Figura 8 - Vista geral da arquitetura do IntelMQ (adaptado) [24]

Da figura a cima, os principais objetos são:

- *Bots* - estrutura que gere mensagens, começa e acabam ações, como por exemplo, envio e tratamento de mensagens através da linguagem Python. Esta estrutura é o ponto crucial deste trabalho pois será ela a responsável por transformar as mensagens numa nomenclatura que seja compreensiva para um SDI. Existem diversos tipos de bots, cada um com a sua responsabilidade, trocando mensagens entre si como ilustra a Figura 9. Estes encontram-se classificados da seguinte forma:
    - *Coletores* – Estes bots são responsáveis pela recolha de informação de um determinado feed. A informação é então enviada para o Parser.
    - *Parsers* – Recebendo como input a informação recolhida pelo coletor, este bot cria um ou mais eventos (dependendo da informação recolhida) respeitando a ontologia do IntelMQ para servir de input aos Experts.
    - *Experts* – Estes processam a informação recebida pelos Parsers e podem executar comandos sobre ela de modo a obter ainda mais informação ou despoletar novas ações. O resultado é então enviado para o Output bot.
    - *Outputs* – Gera informação para o utilizador ou outra ferramenta.

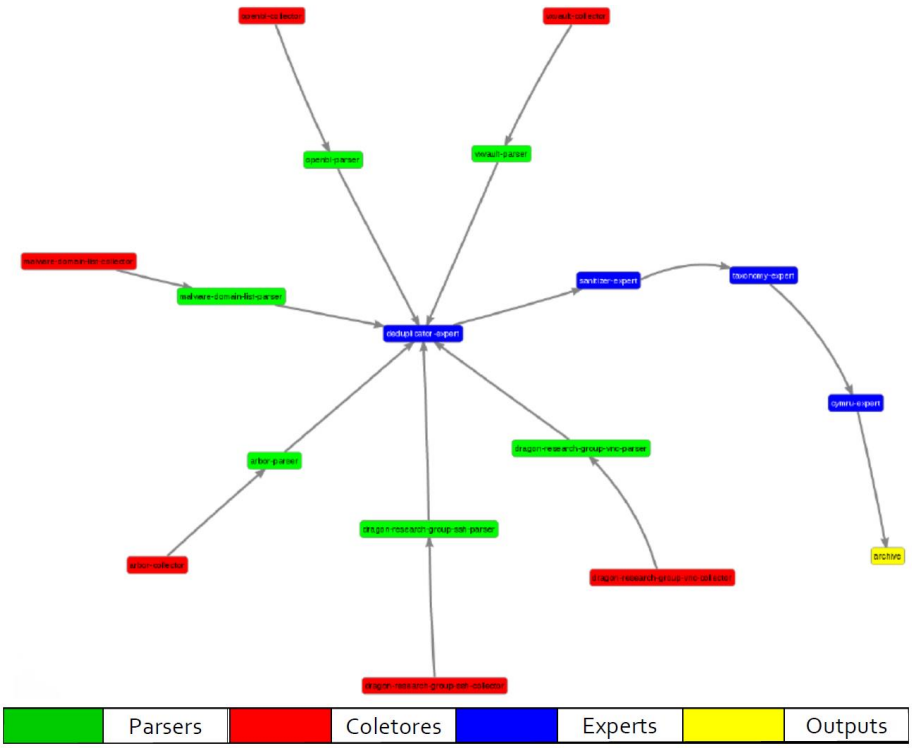

Figura 9 - Esquema de bots do IntelMQ (adaptado) [24]

- *Cache* – Para alguns bots Expert, de modo a agilizar o seu processamento, é utilizada memória cache para armazenar resultados de pesquisas externas. Este armazenamento é suportado pelo Redis.
- *Redis* – É uma estrutura de dados 'em-memória' opensource usada como base de dados, cache e tradutor de mensagens entre interlocutores.
- *Pipeline* – Escreve as mensagens para as filas de mensagens.

O fluxo de dados do IntelMQ é apresentado na Figura 10. O fluxo começa pelos *coletores*, que são bots responsáveis pela recolha de informação de um determinado feed OSINT, passando essa informação para o *parser*. Foram configurados 44 *coletores* OSINT de diversas entidades e respetivos *parsers*. O *parser* cria um ou mais eventos (dependendo da informação recolhida) para servir de input aos *experts*. O *deduplicator* vai evitar que existam eventos duplicados enquanto o *information expert* vai acrescentar informação como por exemplo o *source.asn* (ver parte direita da figura). Os *experts* processam essa informação e executam comandos sobre a mesma de modo a obter mais informação ou despoletar novas ações.

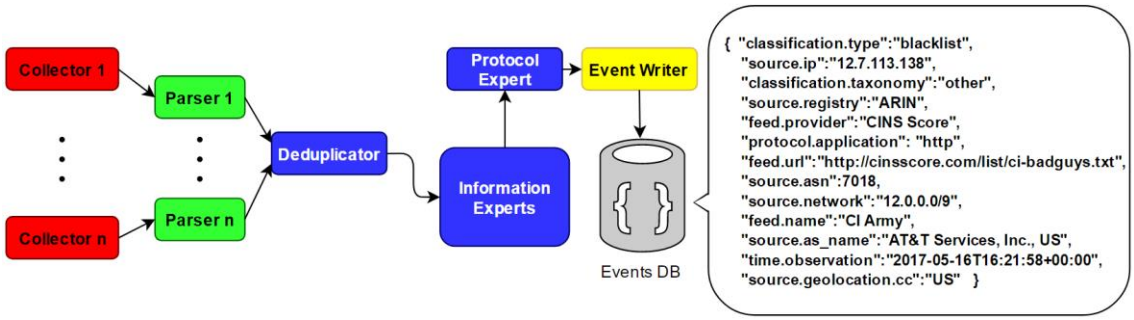

Figura 10 - Configuração IntelMQ

O *protocol expert* é um bot que foi programado de raíz no âmbito deste projeto para permitir identificar no campo *protocol.application* o tipo de protocolo afeto ao evento (ver parte direita da figura). Este campo vem acrescentar informação aos IOCs transformando-os em IOAs, bem como orientar o módulo *Rules & blacklists generator* na criação de regras e na decisão do tipo de regra a gerar. Desta forma as regras criadas vão conter mais informação sobre o evento, o que fará que a regra gere menos falsos positivos.

```
  {
    "rulename": "Abuse.ch BlockList",
    "if": {
      "http_url":
"https\\:\\/\\/feodotracker\\.abuse\\.ch\\/blocklist\\/\\?download\\=d
omainblocklist",
      "protocol.application": ""
    },
    "then": {
      "protocol.application": "dns"
    }
  }
```

Figura 11 - Segmento de código do protocol expert

A Figura 11 mostra um excerto de código do Protocol Expert onde é visível como é gerado o campo protocol.application. Enquanto a *rulename* identifica a regra, o campo *http_url* vai tentar fazer correspondência ao *regex* apresentado e, caso o faça, se o protocol.application ainda não tiver sido preenchido, vai então preenche-lo com o valor dado após o *then*.

Os resultados são então enviados para o event writer, o output bot criado por nós, o qual irá formatá-los, criar novos eventos e armazená-los em events DB.

4.3 Rules & Blacklist Generator

Este módulo foi desenvolvido de raiz no âmbito deste trabalho, para processar os eventos gerados pelo *event writer* e armazenados em *Events DB*, resultando em regras e blacklists para o SDI. O Rule & Blacklist Generator é um script em Python 2.7 que utiliza os IOA guardados (em formato JSON) em *Events* DB gerada pelo *IntelMQ*, para gerar uma regra ou uma entrada blacklist.

A Figura 12 apresenta este módulo em detalhe. Com base na informação contida no campo *protocol.application*, o gerador obtém os protocolos e os portos da regra a gerar. Um exemplo deste funcionamento é a geração de regras através de eventos que designem um domínio malicioso (*protocol.application*="DNS") e que vão permitir a deteção de acesso a esses mesmos domínios por parte de uma máquina na rede. De seguida, o gerador cria a regra com a restante informação contida nos eventos e armazena na base de dados de regras (*Rules DB*). Outro tipo de eventos são os de blacklist, cuja informação é mínima e o único conteúdo relevante é o IP. Estes IPs estão marcados como maliciosos e servem unicamente para gerar uma lista negra para posteriormente alimentar o *preprocessador* de reputação do *Snort*. Desta forma percebe-se se são feitas comunicações para IPs maliciosos, o que pode resultar na contração de vírus e malware por parte dos clientes. Estes IPs são escritos na base de dados blacklist (*Blacklist DB*).

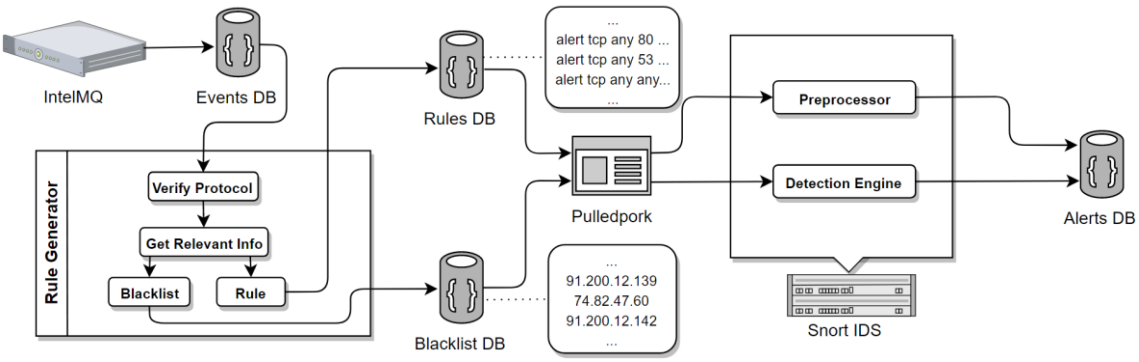

Figura 12- Criação de regras e blacklists a partir dos resultados do IntelMQ

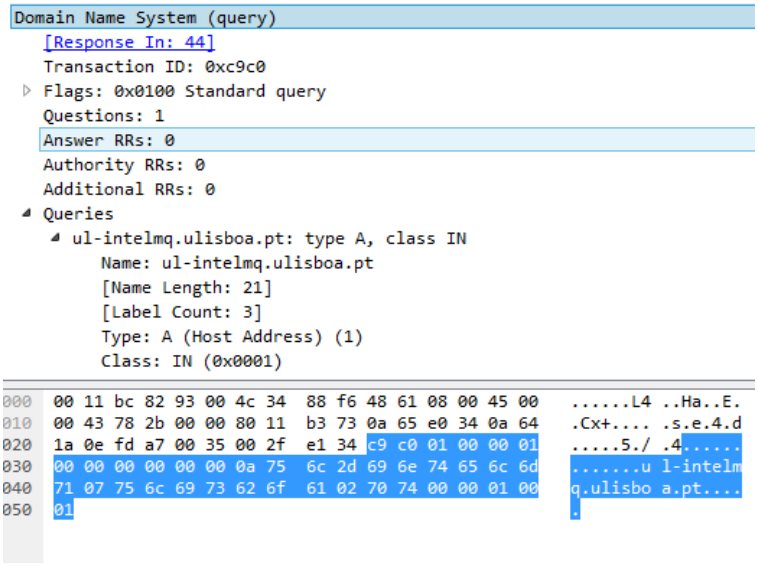

Figura 13 - Query DNS

A Figura 14 mostra como é a estrutura de uma regra gerada pelo Rules & blacklist generator para o Snort e que permite detetar a query DNS apresentada na Figura 13. Detalhadamente cada campo representa o seguinte:

- msg:"…" – Mensagem que é apresentada quando a regra é disparada.
- byte_test: 1,!&,0xF8,2; – Verifica se é um pedido de resolução de domínio, seja ele de que tipo for, recursivo ou iterativo através da comparação de bits.
- fast_pattern; – Verifica primeiro o *content* associado para um aumento de performance. Neste caso, o primeiro *content* pode não ser verificado.
- content "…"; – É o conteúdo a ser pesquisado na *payload* do pacote, que neste caso é o nome *ul-intelmq.ulisboa.pt*.
- nocase; – Torna irrelevante a diferença entre caracteres maiúsculos e minúsculos.
- classtype; – Classe do evento gerado.

```
alert udp $HOME_NET any -> any 53 \
    (msg:"*ul-intelmq.ulisboa.pt domain"; \
     byte_test:1,!&,0xF8,2; \
     content:"|0a|ul-intelmq|07|ulisboa|02|pt|00|"; \
     fast_pattern; \
     nocase; \
     classtype:bad-unknown;)
```

Figura 14 - Estrutura da regra Snort da Query DNS da Figura 13

Outro tipo de eventos são os de blacklist, eventos esses cujo protocol.application é ‘IP’ designa que o único conteúdo relevante é o IP. Estes IPs estão marcados como maliciosos e servem unicamente para gerar uma lista negra para posteriormente alimentar o preprocessador de reputação do Snort. Desta forma percebe-se se são feitas comunicações para IPs maliciosos, o que pode resultar na contração de vírus e malware por parte dos clientes. Estes IPs são escritos na base de dados Blacklist.

4.4 PulledPork

O Pulledpork [55] é um gestor de regras para os SDI Snort e Suricata [32]. O seu funcionamento passa por, de forma automática, recolher regras e IPs de diversas fontes, remover informação duplicada, gerar mapas de regras, gerir reputação de IPs e manter um log de forma cuidada. Estas características tornam-no uma peça fundamental para quem utiliza o Snort.

O Pulledpork é configurado através do ficheiro *pulledpork.con*f onde é descrito o local onde se encontra o ficheiro de regras e de blacklist.

```
rule_url=http://ul-intelmq.ulisboa.pt/rules/blacklist-snort.blf|IPBLACKLIST|open
```

A linha acima indica o caminho onde se encontra o ficheiro com os IPs e que o ficheiro é um ficheiro de blacklist.

```
rule_url=http://ul-intelmq.ulisboa.pt/rules/|regras.tar.gz|regras
```

Esta linha indica onde se encontra o ficheiro com regras e o nome da pasta que se encontra no tarball, que neste caso se chama ‘regras’.

Desta forma o Pulledpork recolhe a informação e gere-a de acordo com a existente e com as restantes fontes de informação (caso existam), e agrega-as nos ficheiros *download.rules* e *black_list.rules*. O ficheiro *download.rules* contém todas as regras recolhidas de todas as fontes enquanto o *black_list.rules* contém todos os IPs recolhidos.

Esta ferramenta realiza toda a gestão de regras da solução e, conjuntamente com uma tarefa no *crontab*, atualiza todos os dias às 00:00 as regras e os IPs blacklist do Snort de forma automática. Isto irá permitir que o SDI se mantenha atualizado sem qualquer tipo de intervenção humana, com base na informação criada pelo Rule Generator.

4.5 SDI Snort e blacklists

As blacklists são listas de IPs com má reputação e conhecidos por fazerem diversos tipos de ações maliciosas como DDoS ou ataques a servidores Apache. O Snort reconhece ficheiros com estas listas e permite utilizá-las no seu preprocessador de reputação para identificar se houve algum tipo de comunicação por parte destes IPs. Esta funcionalidade foi criada a pensar na performance sendo que o seu custo computacional é mais baixo comparativamente ao processador de alertas. Características como estas, bem como a sua simplicidade de utilização, tornaram o Snort o SDI selecionado para integrar a solução.

Capítulo 5

Avaliação experimental

A avaliação da solução proposta foi efetuada em ambiente real, na Reitoria da Universidade de Lisboa, na análise do tráfego da rede interna, durante 8 dias. Foram analisadas as redes do Datacenter, Serviços Centrais da Reitoria (utilizadores) e eduroam.

O capítulo apresenta o set-up da avaliação e o seu comportamento a nível de performance, quais as redes que fizeram parte da captura de tráfego e como estas se encontram interligadas. Na avaliação é dada uma visão sobre os incidentes que foram registados, como se enquadram na informação recolhida e a contabilização dos alertas advindos das regras e blacklists gerados pela solução.

5.1 Set-up e performance da solução

A máquina utilizada na avaliação experimental foi um HP ProLiant DL360 G6 com 2 processadores Intel Xeon CPU X5550 a 2.67GHz, 12Gb de RAM, 165GB de disco rígido e 3 interfaces de rede, sendo uma das interfaces para ligação de fibra ótica a 10Gb/s e as restantes duas de ethernet a 1Gb/s. Foi instalado na máquina o sistema operativo Security Onion, uma distribuição Linux munida de diversas ferramentas de segurança e análise, entre elas o Snort. O IntelMQ foi instalado numa máquina virtual com 4 Gb de RAM e 17 GB de armazenamento.

De facto, só quando uma solução é colocada em produção é que se consegue validar, seja em configuração, qualidade ou performance, e apurar o quanto é necessário afinar [14]. O set-up da solução não fugiu a esta regra, verificando-se que o hardware da solução era insuficiente para aguentar as 3 redes. Face a este problema foi necessário a manutenção constante para possibilitar a recolha dos dados obtidos da melhor forma. Contudo, a percentagem de pacotes perdidos não ultrapassou em média os 0.1%, sendo

que o pior valor de perda atingido foi 5%, verificado em hora de ponta, tendo o sistema aproximadamente 12.000 regras e 16.500 IPs em funcionamento.

5.2 Rede da Universidade de Lisboa

Sendo o cliente da solução, a rede da Universidade de Lisboa vai servir de plataforma onde a solução foi instalada, sendo então necessário uma compreensão aprofundada desta rede.

A universidade é formada por 18 escolas, 11 cantinas, 19 residências e conta com 48.066 alunos inscritos no ano 2014/15 [56]. A infraestrutura de rede da Universidade é portanto, extensa, complexa e com um número muito grande de clientes, impossibilitando toda a sua análise. Posto isto, a solução tratou apenas o tráfego proveniente dos Serviços Centrais, eduroam e Data Center.

Os Serviços Centrais albergam todos os núcleos que contribuem para o bom funcionamento da Reitoria, coordenando assim toda a universidade. O tráfego deste serviço é pouco volumoso, contando com aproximadamente 400 computadores, sendo que nunca se encontram todos simultaneamente ligados.

Nesta rede, os computadores encontram-se protegidos com algumas políticas de segurança e os utilizadores estão cientes de alguns dos perigos que a má utilização da Internet carrega, esperando-se portanto, um número baixo de casos problemáticos. O tráfego que se encontra nesta rede é maioritariamente de trabalho, como por exemplo o acesso a recursos internos como as pastas partilhadas e servidores de e-mail.

A visão geral da rede da UL é apresentada na Figura 15.

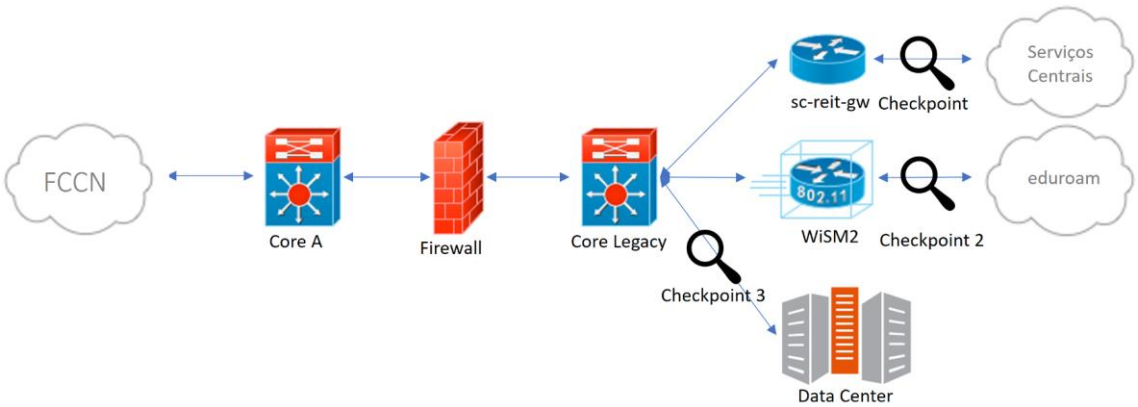

Figura 15 - Esquema lógico de rede da Universidade de Lisboa

A FCCN é fornecedor de acesso à Internet, sendo que é por onde sai todo o tráfego da instituição. O Core A, Firewall e Core Legacy representam pontos intermédios e não têm qualquer papel na análise do tráfego, servindo apenas como meio de transporte de dados.

O sc-reit-gw é o gateway da rede dos Serviços Centrais e é por lá onde passa todo o tráfego administrativo, o WiSM2 é o módulo responsável por todo o tráfego proveniente da eduroam, e o Data Center é onde estão todos os serviços informáticos alojados em servidores físicos e virtuais.

As zonas de ação da solução estão representadas na figura como Checkpoint 1, 2 e 3.

A eduroam é uma rede wireless que disponibiliza à comunidade académica europeia um serviço de mobilidade entre as instituições universitárias participantes. Esta rede encontra-se disponível a alunos, docentes, investigadores e funcionários, sendo possível configurá-la em qualquer dispositivo dotado com ligação de rede sem fios [57] [58]. Isto significa que, qualquer utilizador com credenciais válidas para a rede eduroam, consegue autenticar-se na rede da UL e utilizar os seus serviços.

Com um grupo tão vasto de utilizadores, qualquer equipamento móvel ou computador infetado pode ligar-se a esta rede, podendo então utiliza-la como recurso para tomar ações maliciosas. Esta rede não é vigiada nem controlada sendo um alvo aliciante para criação de "botnets of things".

O tráfego desta rede tem como ponto de comunicação a controladora Wireless da CISCO (WiSM, Wireless Service Modulo) cujo tráfego é posteriormente passado ao Core Legacy como é ilustrado na Figura 15.

O Data Center alberga um número muito grande de serviços como por exemplo e-mail, sites da UL e respetivas Unidades Orgânicas, plataforma SAP, serviços internos de gestão, entre outros. Esta rede contém equipamentos muito sofisticados e potentes para fornecer os serviços sendo então um alvo apetecível.

A decisão de apenas analisar estas três redes provém da falta de recursos computacionais para a análise de todo o tráfego da instituição. No entanto, estas três fontes de tráfego de rede são suficientes para controlo e análise de tráfego.

5.3 Registo de incidentes

Durante o período de avaliação da solução foram usados dois conjuntos distintos de regras e blacklists. Nos primeiros 3 dias, onde decorreu o teste da solução, foram utilizados os mesmos conjuntos de regras e blacklists. Nos restantes 5 dias foram sendo geradas novas regras e blacklists todos os dias, e atualizado o Snort. Durante o tempo em que a solução esteve a recolher resultados, foram geradas em média 5.290 regras por dia e 14.590 entradas de blacklist.

Foram registados aproximadamente 5.5 milhões de incidentes. A Tabela 2 apresenta este valor estratificados pelas diversas categorias de incidentes. O tráfego gerado por IPs pertencentes a blacklists totaliza aproximadamente 70% dos incidentes. Este resultado pode advir do facto do preprocessador não permitir mais nenhum tipo de processamento após haver uma correspondência na blacklist.

Os eventos registados pelo processador das regras correspondem aos restantes 30% do total dos incidentes. Destes, a comunicação SSH por um IP considerado malicioso foi a categoria que registou mais alertas. Na categoria da comunicação com domínios maliciosos foi detetada padrões que possivelmente eram referentes a bots ou C&C, contudo, dada a larga quantidade de dados não foi humanamente possível confirmar se eram ou não de facto. O mesmo sucedeu com os alertas de IMAP, SMTP e FTP. Alguns dos pedidos SIP OPTIONS foram verificados e provinham maioritariamente de uma botnet denominada Friendly-scanner [59] que utiliza como recurso a ferramenta SIPVicious [60]. O mesmo foi verificado nos pedidos de SIP REGISTER. Os restantes alertas foram gerados por comunicações que de IPs maliciosos. Alguns pedidos SNMP foram também confirmados, onde foi possível verificar pedidos realizados com os comandos get e get-next, cujo objetivo é recolher informação detalhada de equipamentos.

Evento	Nº Alertas
Tráfego gerado por um IP em Blacklist	3.779.638
Comunicação SSH por parte de um IP malicioso	1.575.251
Comunicação com Domínio malicioso	45.081
Pedido SIP OPTIONS por IP malicioso	22.126
Comunicação de IP malicioso a servidores WEB	11.770
Comunicação IMAP por parte de um IP malicioso	7.230
Comunicação SMTP por parte de um IP malicioso	4.775
Comunicação FTP por parte de um IP malicioso	1.536
Pedido SIP REGISTER por IP malicioso	1.328
Comunicação Telnet por parte de um IP malicioso	1.003
Comunicação SNMP por parte de um IP malicioso	791
Comunicação SIP por parte de um IP malicioso	369
Total	**5.450.898**

Tabela 2 - Registo de incidentes por categorias de ameaças.

A Tabela 3 mostra o número total de alertas gerados por cada um dos 8 dias. Em média, foram gerados aproximadamente 540.000 alertas nos sete primeiros dias. No último dia houve um aumento significativo de alertas, que possivelmente é justificado com o dia em que surgiu um surto de ransomware Petya [61].

Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Total
410.795	523.005	411.344	505.541	472.309	840.062	605.033	1682.809	**5.450.898**

Tabela 3 - Número total de alertas por dia

Para colmatar o problema de pouca informação gerada pelas regras de blacklist, foi gerada a Tabela 4 com informação que permite identificar as portas de origem e destino dos pacotes blacklist que permite entender o objetivo das comunicações.

Os portos de origem são maioritariamente dinâmicos com a exceção do 6000 e do 0, que coincidem respetivamente com o sistema X11 bem como alguns tipos de trojans [62] e com um porto considerado proibido, mas que é utilizado para recolha de informação [63]. Os portos de destino permitem entender que o serviço mais acedido pelos IP em blacklist é o Telnet (23) seguidamente do porto utilizado pelo serviço MySQL (1433) e pelo SSH (22). Curiosamente o quarto porto mais acedido é um porto utilizado dinamicamente, o que pode revelar que uma máquina foi comprometida e está a comunicar com um IP de blacklist. Os restantes portos mais acedidos foram os de comunicações VoIP (5060), SSDP (1900), HTTP (80), IES (1443) e HTTPS (443). Estes resultados demonstram que os IPs em blacklist devem estar bloqueados em qualquer instituição devido à quantidade de acessos indevidos a serviços como o SSH e o Telnet.

Porto de origem	Nº Alertas	Porto de destino	Nº Alertas
62572	126.073	23	568.242
0	125.324	1433	469.893
10000	75.689	22	386.229
55763	56.497	43526	125.788
52638	44.592	0	125.315
53671	42.513	5060	108.238
65535	35.921	1900	95.962
58022	35.411	80	87.001
52925	30.241	1443	81.796
6000	23.158	443	70.105
Total	**59.5419**	**Total**	**2.118.569**

Tabela 4 - Contagem total de portos de eventos blacklist

A Tabela 5 representa a quantidade de alertas que foram gerados tendo em conta a informação recolhida por cada tipo de categoria OSINT. Desta forma percebe-se quais as categorias de alertas mais geradas na instituição.

Evento OSINT	Nº de Alertas
Blacklist IP	3779638
IPs que tentam obter acesso remoto	1588024
Domínio Malicioso	45081
IPs que atacam serviços de VoIP	23823
IPs que atacam servidores de EMAIL	12005
IPs que atacam servidores FTP	1536
IPs que efetuam scans através de SNMP	791
Phishing ou links perigosos	0
Total	**5450898**

Tabela 5 - Número de alertas gerados por categoria OSINT

A tabela abaixo representa o número total de regras e IPs blacklisted geradas no espaço de 8 dias. Através da correlação destes dados com os da Tabela 5 é percetível que não existe uma ligação entre o número de regras geradas e o número de alertas gerados por categoria, sendo um exemplo disso, o número de alertas de Phishing. Uma razão que pode levar à ausência de alertas da categoria de "Phishing ou links perigosos" é que o IP que leva a cabo este tipo de ataques, tem o IP representado na base de dados de Blacklist, não passando do preprocessador do Snort.

Evento OSINT	Nº de IPs e Regras geradas
Blacklist IP	116313458
Total de IPs	**116313458**
Phishing ou links perigosos	19453
Domínio Malicioso	4223
IPs que tentam obter acesso remoto	864
IPs que atacam servidores de EMAIL	397
IPs que atacam serviços de VoIP	118
IPs que efetuam scans através de SNMP	48
IPs que atacam servidores FTP	23
Total de Regras	**25126**

Tabela 6 - Número de regras e IPs blacklisted por categoria OSINT

Na Figura 16 são visíveis quais os países que geram mais alertas. Esta tabela vai de acordo com a informação do relatório do primeiro quadrimestre de 2017 da Akamai [64]. Países como a China, os Estados Unidos da América e a Holanda que se encontram representados nesse relatório, também constam na figura a baixo.

TOP SOURCE COUNTRIES

COUNT	%TOTAL	#SIG	#DST	COUNTRY	#IP
2177700	43.69%	35	1468	CHINA (.cn)	3011
845412	16.96%	18	2116	UNITED STATES (.us)	1823
300081	6.02%	7	1546	NETHERLANDS (.nl)	311
285193	5.72%	12	1423	FRANCE (.fr)	205
256427	5.14%	12	1446	RUSSIAN FEDERATION (.ru)	437
168943	3.39%	14	1398	KOREA, REPUBLIC OF (.kr)	1093
131749	2.64%	9	1384	INDIA (.in)	928
108663	2.18%	7	1426	UKRAINE (.ua)	409
90225	1.81%	3	1428	SWITZERLAND (.ch)	255
70845	1.42%	8	1384	TAIWAN, PROVINCE OF CHINA (.tw)	632

Figura 16 - Top 10 dos países que originam mais alertas

Também as máquinas da instituição acederam a serviços alojados nos países representados na Figura 17, e que por algum motivo, como por exemplo acesso a um IP blacklisted, foram gerados alertas. Estes dados permitiram descobrir que Portugal se encontra no ranking da lista porque um dos IPs blacklisted que gerou uma grande quantidade de alertas pertencia à instituição, o que pode ter elevado Portugal ao primeiro lugar.

TOP DESTINATION COUNTRIES

COUNT	%TOTAL	#SIG	#SRC	COUNTRY	#IP
3074275	86.79%	149	11537	PORTUGAL (.pt)	1358
252404	7.13%	1	64	ESTONIA (.ee)	45
91139	2.57%	32	2475	UNITED STATES (.us)	1319
23832	0.67%	129	450	CHINA (.cn)	1541
20751	0.59%	28	628	NETHERLANDS (.nl)	90
14436	0.41%	5	143	UNITED KINGDOM (.gb)	63
9888	0.28%	1	234	RUSSIAN FEDERATION (.ru)	215
8851	0.25%	1	437	FRANCE (.fr)	71
8161	0.23%	110	10	BULGARIA (.bg)	47
6394	0.18%	1	438	KOREA, REPUBLIC OF (.kr)	668

Figura 17 - Top 10 dos países de destino que geraram alertas

TOP SOURCE IPS

COUNT	%TOTAL	#SIG	#DST	IP	COUNTRY
824361	15.12%	1	7	61.177.172.52	CHINA (.cn)
252205	4.63%	1	57	10.97.121.91	RFC1918 (.lo)
212296	3.89%	1	7	218.87.109.156	CHINA (.cn)
200798	3.68%	1	3	61.177.172.56	CHINA (.cn)
167704	3.08%	3	1398	213.32.7.73	FRANCE (.fr)
105922	1.94%	1	7	61.177.172.14	CHINA (.cn)
76544	1.40%	1	6373	10.98.106.2	RFC1918 (.lo)
71058	1.30%	1	3	59.63.188.3	CHINA (.cn)
57172	1.05%	1	1372	5.8.50.130	RUSSIAN FEDERATION (.ru)
45277	0.83%	1	1389	80.82.77.33	NETHERLANDS (.nl)

Figura 18 - Top 10 de IPs de origem que geraram alertas

Acima na Figura 18 é representada a quantidade de alertas gerado pelo IP representado, e a que país este IP pertence. Esta figura permite perceber também que existe um acesso anormal deste IP a um conjunto total de máquina representado na coluna *#DST*. Caso o IP esteja a efetuar um Scan à rede, é possível que alguns dos destinos contados na coluna *#DST* não existam.

Nesta lista constam também IPs internos que geraram alertas por comunicarem com IPs maliciosos.

TOP DESTINATION IPS

COUNT	%TOTAL	#SIG	#SRC	IP	COUNTRY
254605	4.67%	9	202	10.103.11.120	RFC1918 (.lo)
251295	4.61%	1	3	5.157.7.179	ESTONIA (.ee)
249152	4.57%	18	1471	10.100.16.22	RFC1918 (.lo)
201341	3.69%	17	1379	10.100.16.25	RFC1918 (.lo)
191261	3.51%	15	1405	10.100.16.24	RFC1918 (.lo)
156000	2.86%	14	1645	10.117.6.240	RFC1918 (.lo)
143311	2.63%	22	1479	10.100.16.23	RFC1918 (.lo)
133358	2.45%	19	1493	10.100.16.21	RFC1918 (.lo)
121892	2.24%	15	1506	10.117.6.95	RFC1918 (.lo)
109561	2.01%	14	233	10.100.209.10	RFC1918 (.lo)

Figura 19 - Top 10 de IPs de destino que geraram alertas

Na Figura 19 estão representados os principais IPs alvo que mais alertas geraram. Esta tabela é útil para perceber quais os serviços mais expostos a tráfego malicioso, tornando fácil de perceber que há uma grande probabilidade de existir algo de errado com

a máquina representada pelo IP. Existe também um acesso anormal a um IP na Estónia considerado malicio que poderá estar comprometido por alguma razão.

Em suma, os dados apresentados nesta secção permitem que uma equipa de segurança tome uma ação para perceber os motivos pelos quais existem estes acessos por estes IPs considerados de maliciosos. Desta forma é possível perceber se existe uma máquina comprometida, ou até más configurações a nível de firewall que permitam acessos indevidos a serviços.

Capítulo 6

Conclusão e trabalho futuro

Este capítulo retrata uma conclusão ao resultado do trabalho realizado e uma proposta de trabalho futuro como forma de continuidade aos resultados obtidos desta dissertação.

6.1 Conclusão

Este estudo teve como intenção entender o paradigma atual do cibercrime no mundo, quais as ameaças que estão relacionadas a estes crimes e como prevenir estas situações. Foi dada também uma visão das ameaças, como operam e se organizam, e o estado da arte dos mecanismos que têm sido desenvolvidos para colmatar este problema.

Foram estudadas diversas técnicas de deteção de intrusões como a host-based e a network-based e suas derivadas, para entender qual a melhor forma de detetar este problema em organizações.

Através deste estudo foi possível desenvolver uma arquitetura de melhoramento para deteção de incidentes por um sistema detetor de intrusões (SDI), baseado em conhecimento extraído de eventos de segurança OSINT. A arquitetura é composta por três componentes, que combinadas, permitem a extração de conhecimento OSINT para geração de novo conhecimento sob as regras de SDI e blacklists, e a deteção de incidentes usando este conhecimento.

A arquitetura foi implementada e avaliada em ambiente real, na Reitoria da Universidade de Lisboa, mostrando-se efetiva no registo de incidentes.

Os resultados permitiram detetar falhas de segurança em serviços tendo sido necessárias algumas intervenções a nível de firewall, demonstrando que a arquitetura apresentada é útil e apresenta resultados positivos.

6.2 Trabalho futuro

Dada a modularidade da arquitetura, é possível portá-la para os restantes SDI, adicionar outras fontes de informação relevantes para serem tratadas automaticamente ou ainda acrescentar novos componentes que permitam adicionar ou melhorar a informação que chega ao SDI.

O MISP [46] pode vir a ter uma relação muito forte com o sistema proposto pois o IntelMQ, parte integrante desta arquitetura, tem um módulo que pode permitir a comunicação com esta ferramenta. Desta forma será possível utilizar o MISP como um feed OSINT, tornando-o um módulo desta arquitetura, sendo até possível utiliza-lo como unidade gestora de informação para o SDI.

A arquitetura desenvolvida pode ser aplicada também a outros tipos de detetores de intrusão como os host-based, que permitirá solidificar os conhecimentos abrangidos por detetores deste tipo, e ainda trocar informação com um detetor network-based, facilitando a comunicação entre ambos.

A partilha de mais conhecimento e mais detalhado sobre incidentes permite potenciar esta arquitetura reduzindo o número de falsos positivos bem como o de falsos negativos, sendo um ponto de começo para alimentar conhecimento em sistemas deste tipo. Isto seria possível através de uma base de dados unificada e comunitária onde se partilham eventos de segurança e informações necessárias para detetar os indicadores das ameaças que o causaram.

Bibliografia

[1] I. Shakeel, “Evolution in the World of Cyber Crime,” InfoSec Institute, 28 Junho 2016. [Online]. Available: http://resources.infosecinstitute.com/evolution-in-the-world-of-cyber-crime/#gref. [Acedido em 1 Dezembro 2016].

[2] N. Gamer, “The state of botnets in late 2015 and early 2016,” TrendMicro, 17 Dezembro 2015. [Online]. Available: http://blog.trendmicro.com/the-state-of-botnets-in-late-2015-and-early-2016/. [Acedido em 1 Dezembro 2016].

[3] Akamai, “Akamai's [state of internet] / security,” Akamai, Massachusetts, 2016.

[4] “Mirai VS Nitol Infographic,” 12 1 2017. [Online]. Available: https://www.incapsula.com/blog/wp-content/uploads/2016/12/MiraivsNitol_Infographic.final_.jpg. [Acedido em 6 Janeiro 2017].

[5] N. Woolf, “DDoS attack that disrupted internet was largest of its kind in history, experts say,” 12 1 2017. [Online]. Available: https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet. [Acedido em 5 Janeiro 2017].

[6] P. Loshin, “Details emerging on Dyn DNS DDoS attack, Mirai IoT botnet,” 12 1 2017. [Online]. Available: http://searchsecurity.techtarget.com/news/450401962/Details-emerging-on-Dyn-DNS-DDoS-attack-Mirai-IoT-botnet. [Acedido em 5 Janeiro 2017].

[7] D. P. A. S. Mark Kraynak, “Cyber Security Predictions for 2016,” 2015. [Online]. Available: http://blog.imperva.com/2015/12/5-cyber-security-predictions-for-2016.html. [Acedido em 12 Janeiro 2017].

[8] A. Shulman, “Trends IT Security Pros Need to Think About Going into 2017 and What to Do About Them,” 2016. [Online]. Available: http://blog.imperva.com/2016/12/3-trends-it-security-pros-need-to-think-

about-going-into-2017-and-what-to-do-about-them-.html. [Acedido em 12 Janeiro 2017].

[9] S. Morgan, "Cyber Crime Costs Projected To Reach $2 Trillion by 2019," Forbes, 17 Janeiro 2016. [Online]. Available: https://www.forbes.com/sites/stevemorgan/2016/01/17/cyber-crime-costs-projected-to-reach-2-trillion-by-2019/. [Acedido em 25 Junho 2017].

[10] J. M. Butle, *Finding Hidden Threats by Decrypting SSL,* SANS Institute, 2013.

[11] M. Rouse, "Metamorphic and polymorphic malware," SearchSecurity, Outubro 2010. [Online]. Available: http://searchsecurity.techtarget.com/definition/metamorphic-and-polymorphic-malware. [Acedido em 5 Dezembro 2016].

[12] G. Bruneau, *The History and Evolution of Intrusion Detection,* SANS Institute, 2001.

[13] S. Silva, R. Silva, R. Pinto e R. Salles, "Botnets: A Survey," *Elsevier,* pp. 379-399, 2012.

[14] A. H. Adam J. Aviv, "Challenges in Experimenting with Botnet Detection Systems," em *CSET*, 2011.

[15] T. Hoff, "Scaling Twitter: Making Twitter 10000 Percent Faster," High Scalability, 27 Junho 2009. [Online]. Available: http://highscalability.com/scaling-twitter-making-twitter-10000-percent-faster. [Acedido em 19 Novembro 2016].

[16] P. Bächer, T. Holz, M. Kötter e G. Wicherski, "Know your Enemy: Tracking Botnets," Honeynet, [Online]. Available: https://www.honeynet.org/book/export/html/50. [Acedido em 30 Novembro 2016].

[17] I. Vacas e I. Medeiros, "Geração Automática de Conhecimento para SDI extraíido de OSINTs," em *9th INForum - Simpósio de Informática (INForum'17),* Aveiro, Portugal, 2017.

[18] M. Rahimipour e S. Jamali, "A Survey on Botnets and Web-based Botnet Characteristics," *International Journal of Computer Science Engineering and Technology,* pp. 282-285, 2014.

[19] R. Puri, *Bots & Botnets: An Overview,* SANS Institute, 2003.

[20] Team, TrendLabs APT Research, "Spear-Phishing Email: Most Favored APT Attack Bait," Trend Micro, 2012. [Online]. Available: http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-spear-phishing-email-most-favored-apt-attack-bait.pdf. [Acedido em 10 Dezembro 2016].

[21] Symantec, "Bots and Botnets - A Growing Threat," [Online]. Available: https://us.norton.com/botnet/. [Acedido em 20 Dezembro 2016].

[22] MalwareTech, "Peer-to-Peer Botnets for Beginners," MalwareTech, 22 Dezembro 2013. [Online]. Available: https://www.malwaretech.com/2013/12/peer-to-peer-botnets-for-beginners.html. [Acedido em 1 Dezembro 2016].

[23] T. Matthews, Incapsula Survey: What DDoS Atacks Really Cost Businesses, Incapsula, 2014.

[24] T. Lewis, *HTTP header heuristics for malware detection,* SANS Institute, 2013.

[25] "Appendix A: Botnet Commands - Which commands the bots understand," Honeynet, 8 Outubro 2008. [Online]. Available: https://www.honeynet.org/node/55. [Acedido em 16 Dezembro 2016].

[26] M. Pradhan, C. Kayak e S. Pradhan, Network Security Attacks and Countermeasures, IGI Global, 2016.

[27] C. K. Giovanni Vigna, *Host-Based Intrusion Detection,* Citeseer, 2005.

[28] Snort, "Snort," Snort, [Online]. Available: https://www.snort.org. [Acedido em 21 Janeiro 2017].

[29] D. Dineley, "The greatest open source software of all time," InfoWorld, 17 Agosto 2009. [Online]. Available: http://www.infoworld.com/article/2631146/open-source-software/the-greatest-open-source-software-of-all-time.html. [Acedido em 21 Janeiro 2017].

[30] M. J. d. l. Merced, "Cisco to Buy Sourcefire, a Cybersecurity Company, for $2.7 Billion," DealBook, 23 Julho 2013. [Online]. Available: https://dealbook.nytimes.com/2013/07/23/cisco-to-buy-sourcefire-a-

cybersecurity-company-for-2-7-billion/?_r=0. [Acedido em 21 Janeiro 2017].

[31] N. Dietrich, "The Reputation Preprocessor in Snort – Blacklists and Whitelists," Snort, Technology, 10 12 2015. [Online]. Available: https://sublimerobots.com/2015/12/the-snort-reputation-preprocessor/. [Acedido em 01 06 2017].

[32] "Suricata," Suricata, [Online]. Available: https://suricata-ids.org/. [Acedido em 21 Janeiro 2017].

[33] "The Bro Network Security Monitor," Bro, [Online]. Available: https://www.bro.org/. [Acedido em 21 Janeiro 2017].

[34] R. McCarty, "Network analysis with the Bro Network Security Monitor," Admin-Magazine, 2014. [Online]. Available: http://www.admin-magazine.com/Archive/2014/24/Network-analysis-with-the-Bro-Network-Security-Monitor. [Acedido em 21 Janeiro 2017].

[35] M. Kershaw, "Documentation," KismetWireless, 01 2016. [Online]. Available: https://www.kismetwireless.net/documentation.shtml. [Acedido em 21 Janeiro 2017].

[36] "OSSEC - Open Source HIDS SECurity," OSSEC, 2017. [Online]. Available: http://ossec.github.io/. [Acedido em 21 Janeiro 2017].

[37] Mike, "The SAMHAIN file integrity / host-based intrusion detection system," Samhain Labs, 2006. [Online]. Available: http://www.la-samhna.de/samhain/. [Acedido em 21 Janeiro 2017].

[38] G. Hribar, I. Podbregar e T. Ivanuša, "OSINT: A "Grey Zone"?," *International Journal of Intelligence and CounterIntelligence,* vol. 27, pp. 529-549, 2014.

[39] Q. Eijkman e D. Weggemans, "Open Source Intelligence and Privacy Dilemmas: Is It Time to Reassess State Accountability?," *Sec. Hum. Rts.,* vol. 23, p. 285, 2012.

[40] A. Compton, "British Tourists Detained, Deported For Tweeting 'Destroy America'," Huffingtonpost, 30 Janeiro 2012. [Online]. Available: http://www.huffingtonpost.com/2012/01/30/british-tourists-deported-for-tweeting_n_1242073.html. [Acedido em 20 Julho 2017].

[41] “Phishtank,” OpenDNS, [Online]. Available: https://www.phishtank.com/. [Acedido em 20 6 2017].

[42] “The CINS Score,” Sentinel IPS, [Online]. Available: http://cinsscore.com/. [Acedido em 20 Junho 2017].

[43] Y. Zeng, X. Hu e K. Shin, “Detection of botnets using combined host-and network-level information. In Dependable Systems and Networks (DSN),” em *IEEE/IFIP International Conference on (pp. 291-300)*, 2010.

[44] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras e B. Stiller, “An over-view of ip flow-based intrusion detection,” IEEE Communications Surveys and Tutorials, 2010.

[45] Z. Zhu e T. Dumitras, “FeatureSmith: Automatically Engineering Features for Malware Detection by Mining the Security Literature,” em *2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[46] C. Wagner, A. Dulaunoy, G. Wagener e A. Iklody, “MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform,” em *2016 ACM on Workshop on Information Sharing and Collaborative Security (pp. 49-56)*, 2016.

[47] G. Guofei, J. Zhang e L. Wenke, “BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic.,” em *NDSS*, 2008.

[48] G. Fedynyshyn, M. Chuah e G. Tan, “Detection and classification of different botnet C\&C channels,” *Autonomic and trusted computing,* pp. 228-242, 2011.

[49] J. Goebel e T. Holz, “Rishi: Identify Bot Contaminated Hosts by IRC Nickname Evaluation.,” *HotBots,* vol. 7, p. 8, 2007.

[50] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten e I. Osipkov, “Spamming botnets: signatures and characteristics,” *ACM SIGCOMM Computer Communication Review,* vol. 38, pp. 171-182, 2008.

[51] P. Wurzinger, L. Bilge, T. Holz, J. Goebel, C. Kruegel e E. Kirda, “Automatically generating models for botnet detection,” em *European symposium on research in computer security,* 2009.

[52] N. Lord, “What are indicators of compromise?,” Digital Guardian, 27 Julho 2017. [Online]. Available: https://digitalguardian.com/blog/what-are-indicators-compromise. [Acedido em 27 Julho 2017].

[53] Intel Security, “Indicators od Attack (IoA),” Intel Security, Santa Clara, CA, 2014.

[54] “Automate Incident Handling Process : IntelMQ,” n0where, [Online]. Available: https://n0where.net/automate-incident-handling-process-intelmq/. [Acedido em 10 Janeiro 2017].

[55] Shirkdog, “PulledPork,” [Online]. Available: https://github.com/shirkdog/pulledpork. [Acedido em 29 Maio 2017].

[56] “Factos e Números,” Universidade de Lisboa, [Online]. Available: https://www.ulisboa.pt/home-page/universidade/factos-e-numeros/. [Acedido em 10 Janeiro 2017].

[57] “eduroam,” FCCN, [Online]. Available: https://eduroam.pt/pt/sobre/descricao. [Acedido em 12 Janeiro 2017].

[58] “Bem-vindo a rede sem fios da UE,” Universidade de Evora, [Online]. Available: http://wifi.uevora.pt/. [Acedido em 12 Janeiro 2017].

[59] M. Kezys, “SIP Attack: Friendly-Scanner,” 24 Março 2015. [Online]. Available: http://blog.kolmisoft.com/sip-attack-friendly-scanner/. [Acedido em 27 Junho 2017].

[60] S. Gauci, “SipVicious,” [Online]. Available: https://github.com/EnableSecurity/sipvicious. [Acedido em 27 Junho 2017].

[61] A. H. Olivia Solon, “'Petya' ransomware attack: what is it and how can it be stopped?,” The Guardian, [Online]. Available: https://www.theguardian.com/technology/2017/jun/27/petya-ransomware-cyber-attack-who-what-why-how. [Acedido em 28 Junho 2017].

[62] SpeedGuide, “Port 6000 Details,” SpeedGuide, [Online]. Available: https://www.speedguide.net/port.php?port=6000. [Acedido em 27 Junho 2017].

[63] B. Mitchell, “Port 0 in TCP and UDP,” Lifewire, 10 Junho 2017. [Online]. Available: https://www.lifewire.com/g00/port-0-in-tcp-and-udp-818145?i10c.referrer=. [Acedido em 28 Junho 2017].

[64] Akamai, “Akamai's [state of internet] / security,” Akamai, Massachusetts, 2017.

[65] “Nitol Botnet Fuels 8.7 Gbps Layer 7 DDoS Attack,” 2016. [Online]. Available: http://www.securityweek.com/nitol-botnet-fuels-87-gbps-layer-7-ddos-attack. [Acedido em 17 Janeiro 2017].

[66] Waqas, “Meet the Leet DDoS Botnet, Just as Powerful as Mirai,” 2016. [Online]. Available: https://www.hackread.com/meet-the-leet-ddos-botnet-as-powerful-as-mirai/. [Acedido em 12 Janeiro 2017].